Fall 2018

IMPROVING A TRANSIENT STABILITY CONTROL SCHEME WITH WIDE-AREA SYNCHROPHASORS AND THE MICROWECC, A REDUCED-ORDER MODEL OF THE WESTERN INTERCONNECT

Robin Hallett
Montana Tech

IMPROVING A TRANSIENT STABILITY CONTROL SCHEME WITH WIDE-AREA SYNCHROPHASORS AND THE MICROWECC, A REDUCED-ORDER MODEL OF THE WESTERN INTERCONNECT

by

Robin Hallett

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science Electrical Engineering

Montana Technological University

2018

Abstract

This thesis is composed of two research projects. The first project investigated the feasibility of improving a generator tripping control scheme using wide-area synchrophasors and the second project focused on building a reduced-order model of the western North American power system (wNAPS) for use in a real-time digital simulator.

Transient stability is a major reliability issue for power systems. Radially-connected power plants are especially prone to transient stability problems. An example of this is the four-generator Colstrip power plant located in southeast Montana, USA. The Colstrip generators are protected by a tripping control system called the Acceleration Trend Relay (ATR) that is designed to disconnect generators during system disturbances to prevent asynchronous operation and further system instability. Like most transient stability tripping schemes, the ATR relies entirely on local information. However, because transient stability is a wide-area phenomenon determined by the relative synchronism of the system, local information can produce misoperations causing the ATR to false trip. The first part of this thesis studied the feasibility of using wide-area synchrophasors provided by phasor measurement units (PMUs) to improve protection schemes such as the ATR. Transient stability software was used to model the ATR and evaluate the benefits of adding wide-area measurements to the control scheme.

Real-time simulators are effective tools for studying power systems because they can accurately reproduce electromechanical dynamics while allowing for prototype controllers to be physically connected. However, they impose serious limitations on the size of systems that can be modeled. Model order reduction techniques can be used to lower the computational complexity of a system while approximating the dynamics of the original model. The second part of this thesis presents a reduced-order model of the wNAPS termed the “MicroWECC.” The MicroWECC is a further reduction of the MiniWECC model and was designed to have approximate impedance, generation, and modal characteristics. The model was constructed in two positive-sequence transient simulation tools and then modal analysis was performed to compare the MicroWECC to the MiniWECC model. Parameters for electromagnetic transient program (EMTP) models were also suggested.

Keywords: Transient stability, Acceleration Trend Relay, wide-area, phasor measurement unit, synchrophasor, Western Interconnect, reduced-order model, real-time digital simulator, modal analysis, eigenanalysis

Dedication

This thesis is dedicated to my parents, Frank and Melonie. Thank you for supporting and encouraging my academic career.

I would also like to thank all the friends I have had the pleasure of meeting throughout my years at Montana Tech. I would have not pursued this degree without you.

Acknowledgements

I would like to acknowledge my fantastic professors, advisors, and mentors at Montana Tech: Dr. Dan Trudnowski, Dr. Matt Donnelly, Dr. Josh Wold, Dr. Bryce Hill, Dr. John Morrison, Dr. Tom Moon, and Dr. Curtis Link.

I would also like to thank NorthWestern Energy engineers Eric Bahr and Chelsea Loomis for their assistance in this research.

I would also like to acknowledge Bonneville Power Association and NorthWestern Energy for supporting this research.

Table of Contents

List of Tables

List of Figures

List of Equations

Equation

Glossary of Terms

Term	Definition
ATR	Acceleration Trend Relay
BPA	Bonneville Power Association
EMTP	Electromagnetic Transient Program
Hz	Hertz
MW	Mega Watt(s)
NERC	North American Electric Reliability Corporation
PMU	Phasor Measurement Unit
PSCAD	Power Systems Computer-Aided Design
PSLF	Positive Sequence Load Flow
PSS	Power System Stabilizer
PST	Power System Toolbox
PU	Per-Unit
RAS	Remedial Action Scheme
RSCAD	RTDS Simulator Computer-Aided Design
RTDS	Real Time Digital Simulator designed by RTDS Technologies
RTS	Real Time Digital Simulator
sps	Samples per second
SPS	Special Protection System
WECC	Western Electrical Coordinating Council
wNAPS	Western North American Power System

1. Introduction

Modern electric grids are the largest and most complex machines in the world. The western North American power grid (wNAPS), also known as the Western Interconnect, consists of over 100,000 miles of transmission lines that transmit power from hundreds of generators to millions of customers and countless machines.

The wNAPS is a wide-area alternating current electrical grid that operates at a synchronized 60 Hz frequency. This power system also contains special direct current ties to help stabilize areas that have difficulty remaining synchronized. The entire system is electrically tied together during normal operating conditions, however disturbances such as faults and outages negatively affect the overall stability of the grid. The more complex a power system is and the more power that is being transmitted, the more likely the grid is to suffer stability problems.

System stability is essential for delivering reliable power to the industrial, commercial, and residential consumers that depend on the grid. Instability can lead to large-scale power outages, resulting in massive economic consequences and possibly hazardous situations for those affected. Strict requirements and standards are set by the North American Electric Energy Reliability Corporation (NERC) to ensure the grid remains stable during unplanned outages, also referred to as contingencies [1]. These requirements are monitored and enforced by regional entities such as the Western Electricity Coordination Council (WECC), which oversees the reliability of the Western Interconnect [2].

The ability for a power system to remain synchronized during or after a significant contingency is known as transient stability. Transient stability is often the limiting factor in power transmission and a major reliability concern for regional entities. Transient stability

studies are therefore necessary to evaluate the dependability and security of a power system. To perform transient stability studies, software tools are utilized by engineers to build realistic power system models and simulate contingencies that otherwise could not be performed on the physical grid. Transient stability studies can also be used to test and design control schemes to mitigate the effects of critical disturbances.

NERC standards require the use of such control schemes to protect the bulk power system. Depending on the design or operation, these control systems can also be referred to as remedial action schemes (RAS) or special protection systems (SPS). In general, both SPS and RAS systems are sophisticated schemes used to bolster the transient stability of power systems. These protection schemes are designed to monitor system conditions and take corrective actions automatically during abnormal or predetermined events [3]. Actions are established in advance from studies and include switching loads, generation, or the system configuration to stabilize the system [3].

Generation tripping schemes are among the most reliable methods for preserving system stability. An example of such a scheme is the ATR employed at the Colstrip power plant in southeast Montana. This coal-fired generation plant is located on the end of a long radial transmission network in the wNAPS. Radially-connected power plants are especially prone to transient stability problems because during system disturbances, reconfiguration of the power system is limited. Generator trip schemes like the ATR reduce the total power on the system making them particularly effective in stabilizing these systems.

The purpose of the ATR is to protect customers from power outages, prevent cascading events, and avoid WECC violations. It achieves this by detecting disturbances and calculating the appropriate number of generators to trip offline based on the severity of the event. The

additional stability offered by the ATR allows for full generation output at Colstrip and an increased transfer capacity on the Montana 500kV transmission system [4]. The ATR is engineered to handle a large variety of unplanned events and failures in the Montana power system and appropriate trip decisions are based on decades of transient stability studies.

1.1. Problem Statement

The ATR operates entirely on local information. This is considered a major design advantage because the performance of the control scheme does not depend on the reliability of long distant communications [4]. Local information can have its drawbacks, however. In general, transient stability is a wide-area phenomenon established by power-angle relationships between generators [5]. Local information inherently lacks observability of wide-area angle variations and therefore is not a robust indicator of the transient condition [5]. Generator trip schemes that rely entirely on local measurements cannot meet strict reliability requirements without risking unnecessary generation drops in certain cases.

It was hypothesized that augmenting the ATR algorithm with wide-area phasor measurements could provide improved observation of the transient condition and allow the ATR to make more selective trip decisions. The objective of this research was to investigate the feasibility of using wide-area synchrophasors provided by a remote PMU to reduce false trips.

Similar research was performed in [5]. The potential benefits and feasibility of using PMU measurements from Grand Coulee were investigated in this paper. The authors concluded that relative phase between Grand Coulee and Colstrip not only would be a robust indicator of inter-area transient stability, but in steady-state operations, can also serve as a quantitative indicator of the transmission system stress.

1.2. Procedure

To test the hypothesis, the ATR scheme, Colstrip, and the Western Interconnect would first be modeled in transient stability software. A control scheme would then be proposed to augment the existing ATR with wide-area PMU measurements. Finally, the performance of the ATR with the new control scheme would be evaluated during a variety of contingencies. An outline of this procedure is shown in Figure 1.

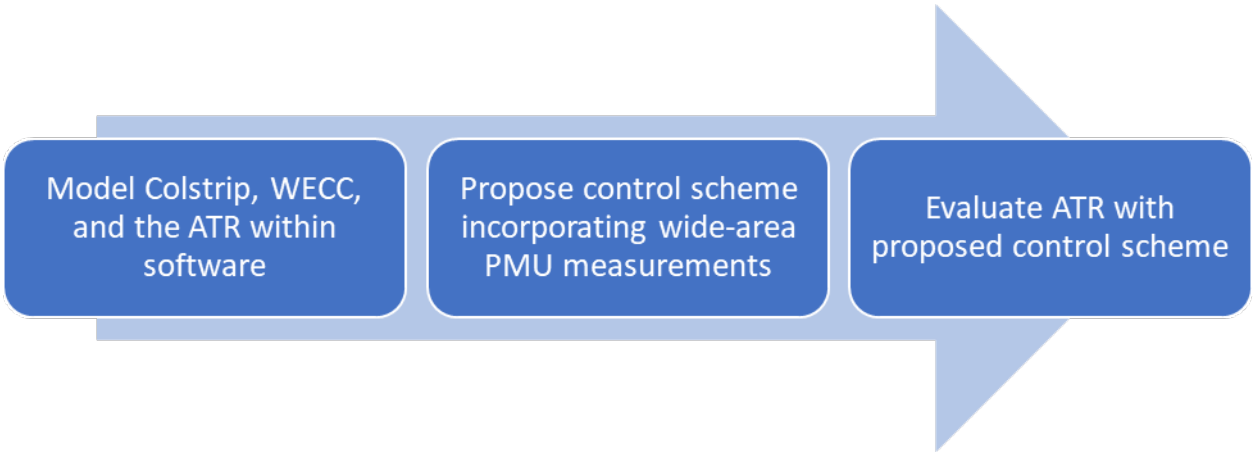

Figure 1. Procedure for Project 1

2. ATR, Colstrip, and WECC Modeling

For the purposes of this research, evaluating the performance of the ATR and simulating wide-area synchrophasor information could not be done directly. Transient stability software was needed to model the wNAPS and the ATR, and to perform studies. Power system analysis programs can be divided into two broad categories; commercial and open-source. Commercial software is typically used when efficiency is required, while open-source is intended for educational or research objectives. Although commercial software packages tend to be better tested and computationally optimized, they often prohibit modifying source code and adding new algorithms. Modeling the ATR control scheme would require adding algorithms to the source code, so the flexibility found in open-source software was necessary. For this reason, the open-source software package Power System Toolbox (PST) was used for all simulations completed in this research project.

2.1. Western Interconnect Model in PST

PST uses a set of MATLAB M-files to model power system components needed for load flow and transient stability simulations. Components include both static models such as transmission lines and loads, as well as dynamic models such as generators, exciters, turbine-governors, and power system stabilizers (PSS). A reduced-order PST model of the Western Interconnect, termed the “MiniWECC” was provided for this research to perform transient stability studies on [6]. This model is a 472nd order model containing 122 buses, 19 load centers, 34 generation areas, and 2 DC lines. The data file for the MiniWECC can be found in Appendix L. A one-line diagram for the model is shown in Figure 2. The Colstrip power plant is labeled as generator 14.

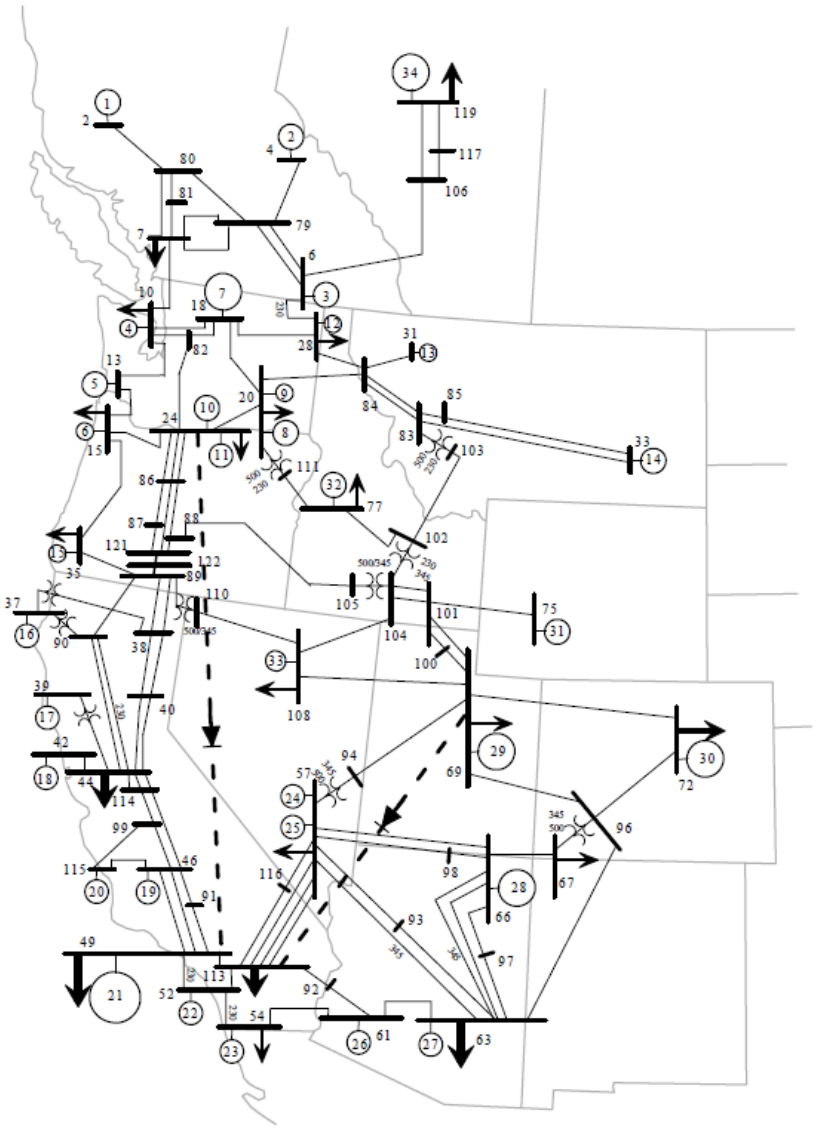

Figure 2. MiniWECC One-Line Diagram

2.2. Colstrip Model in PST

The ATR has inputs from all four generator units; therefore, generator 14 was separated into four individual units in the model. The two larger units were made 2.75 times greater than the two small units in terms of power generation, inertia, and transformers. The combined power output, inertia, and transformer impedance of the four individual units were made equivalent to the original single unit. Exciter, PSS, and turbine-governor models for the individual units were

made the same as the original generator. A dynamic simulation was recorded in both models by faulting the line between buses 83 and 85 for 5 cycles before opening the line. Next, generator speeds were compared to ensure the model had not been changed. The Colstrip generator speeds are shown in Figure 3(a) and speed differences between models are shown in Figure 3(b). Note that in the first plot, the new generators speeds are directly on top of the original generator. In the second plot, note that speed differences are close to machine precision. These plots provide strong evidence that the dynamics of the model were not affected by the changes made to the Colstrip generators.

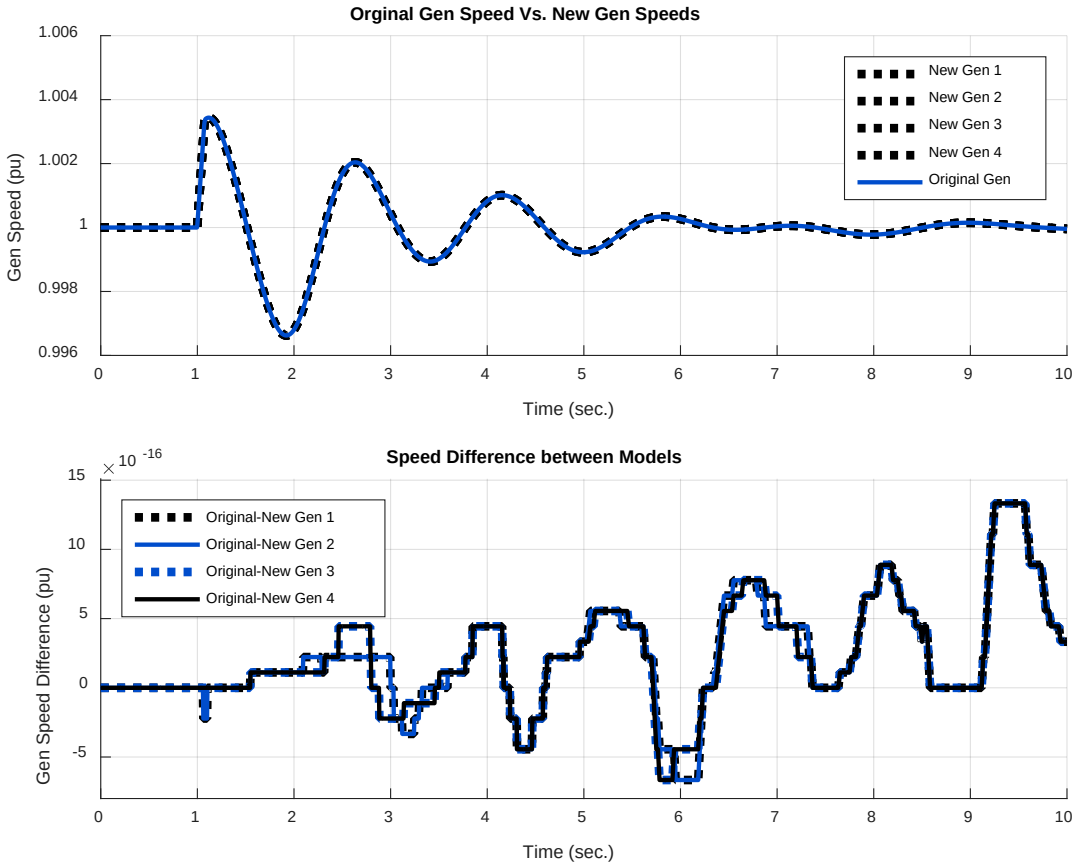

Figure 3. Comparison of Original Gen to New Gens

2.3. ATR Overview

The next task in this project was to model the ATR control scheme. An overall block diagram of the ATR is provided in Figure 4. This diagram illustrates the flow of calculations performed by the ATR starting with the unit calculations sub-block and ending with the unit relaying sub-block.

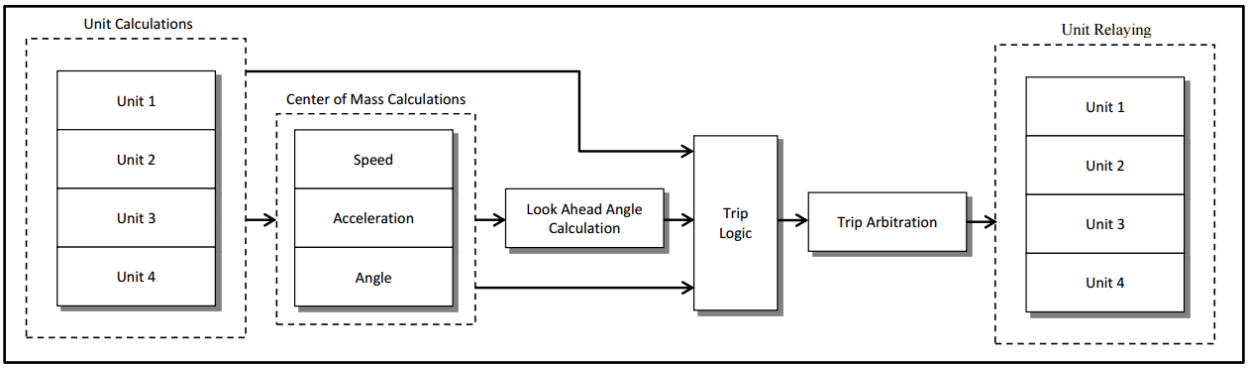

Figure 4. ATR Block Diagram. (Prepared by Eric Bahr)

Unit calculations are performed first using inputs of electric power and speed deviation from the individual generators. This information is used to produce speed, acceleration, and angle values for each generator. As shown in the figure, the information is then passed to the center-of-mass calculations sub-block where the individual speed, acceleration, and angle values are averaged together for the entire plant. A predicted look-ahead angle is calculated and passed, along with the other values, to the trip logic sub-block. If an event is detected by the trip logic, a trip percentage is calculated and passed to the trip arbitration sub-block. This sub-block determines which units to drop to satisfy the percentage. Finally, illustrated by the unit relaying sub-block, a trip signal is sent to individual circuit breakers to disconnect the units.

2.3.1. Individual Unit Calculations

The first section of the ATR block diagram in Figure 4 is the unit calculations sub-block. This is where electrical power and speed deviation enter the ATR. The sample rate of the

transducer inputs are 150 sps [5]. The output of this sub-block is speed, angle, and acceleration for individual units. These are the primary measurements for the trip logic.

2.3.1.1. Speed Calculation

The raw speed of each generator unit is measured by tooth wheels and magnetic pickups. This input passes through two low-pass filters to remove noise and torsional vibrations to produce a smooth speed signal [4]. Next, a washout filter creates the speed deviation signal by rejecting steady-state inputs and passing transients [4].

2.3.1.2. Angle Calculation

The angle value is calculated by numerically integrating speed and is passed through another washout filter to prevent the angle from increasing without bound [4]. The angle calculation is a measure of the relative motion of the shaft from its local constant angle and tends to reach zero when operating at nominal speed [4].

2.3.1.3. Acceleration Calculation

According to [4], the electric power input is measured by watt/var transducers on each generator. Low pass filters are used to remove noise and to produce a slow-responding electrical power signal. A correction signal is added to produce a proxy mechanical power quantity. The difference between the proxy mechanical power and the instantaneous electrical power is taken to produce the accelerating power. The correction signal is created by differentiating the speed input to produce a mechanical acceleration and then subtracting the accelerating power. The result is passed through a wide bandwidth low pass filter to become the correction signal. This feedback loop results in an accelerating power that responds instantaneously to changes in electrical power but over time converges to the mechanical acceleration value [4].

2.3.2. Center of Mass Calculation

The three outputs from the unit calculations are passed to the center-of-mass sub-block. The purpose of this calculation is to average the individual unit values into a single value for the entire plant. A simplified formula for the center-of-mass calculation is shown in (1):

$$CMval = \frac{U1val * U1stat + U2val * U2stat + 2.75(U3val * U3stat + U4val * U4stat)}{U1stat + U2stat + 2.75(U3stat + U4stat)} \quad (1)$$

Where the *UXstat* represents the online status of a generator; 1 for online, 0 for offline, and *UXval* represents either a speed, angle, or acceleration value, where X represents a number, 1 through 4, corresponding to the appropriate generator. A 2.75 gain is applied on units 3 and 4 because their moment of inertia is roughly 2.75 times larger than units 1 and 2 [7].

2.3.3. Look Ahead Angle Calculation

The center-of-mass angle, acceleration, and speed are then used to generate a look-ahead (LA) angle value. The LA angle is the projected angle of the plant 6 cycles into the future, based on a constant acceleration assumption [7]. A simplified formula for calculating the LA angle is shown in (2):

$$lookAheadAng = speed * 15 + accel * 35.971 + angle \quad (2)$$

2.3.4. Trip Logic

The center-of-mass values, look-ahead angle, and individual unit speeds are passed to the trip logic. This section of the ATR consists of 11 individual algorithms that are each designed to detect a specific class of unstable events. Some, like the over-speed algorithm, issue a trip decision when the speed of an individual unit reaches a fixed threshold. Others, like the acceleration-speed algorithm, compare several quantities (speed and acceleration) to determine a

trip percentage. For a detailed explanation of each individual algorithm, the reader is referred to [4] and [7]. It is possible for multiple trip logic algorithms to make a simultaneous trip percentage decision. In these cases, the maximum trip percentage is passed to the trip arbitration.

2.3.5. Trip Arbitration

The trip arbitration logic takes the trip percentage issued from the trip algorithms and decides which units to trip. The units are decided by first calculating the amount of power necessary to trip offline. This is done by multiplying the trip percentage by the amount of online power. The power output of each individual unit is also known. Next the logic decides from a list of all possible trip combinations the preferred combination that is equal to or greater than the submitted call [4]. Preferences can be selected to trip either a small unit over a large unit, a large unit over two small units, a specific large unit, or a specific small unit. If two combinations have equal preference, the one with the lowest combination of power will be chosen [7]. Signals are then sent to individual breakers to disconnect the units.

2.4. Simulink ATR Model

A continuous-time Simulink model of the ATR was provided by NorthWestern Energy engineer Eric Bahr for this research. This model was created by Mr. Bahr to be used with the output of a positive-sequence transient stability software such as PST. The overall Simulink ATR model that was provided is shown in Figure 5. This model does not feature the trip arbitration and unit relaying components.

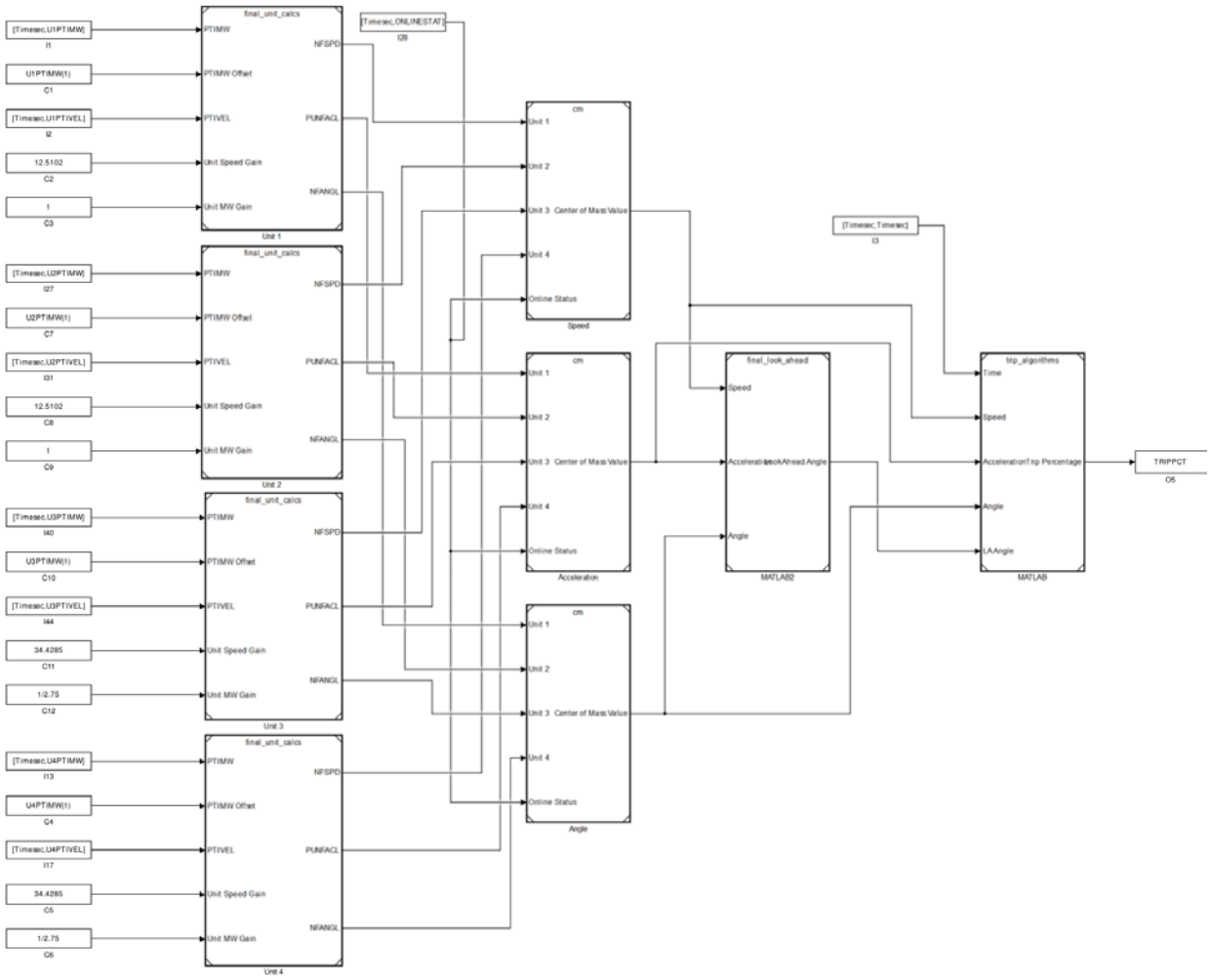

Figure 5. Simulink ATR Block Diagram

2.4.1. Simulink Unit Calculations

Figure 6 illustrates a simplified block diagram prepared by Mr. Bahr for the unit calculations and represents a close approximation of how speed, angle, and acceleration are calculated by the actual ATR.

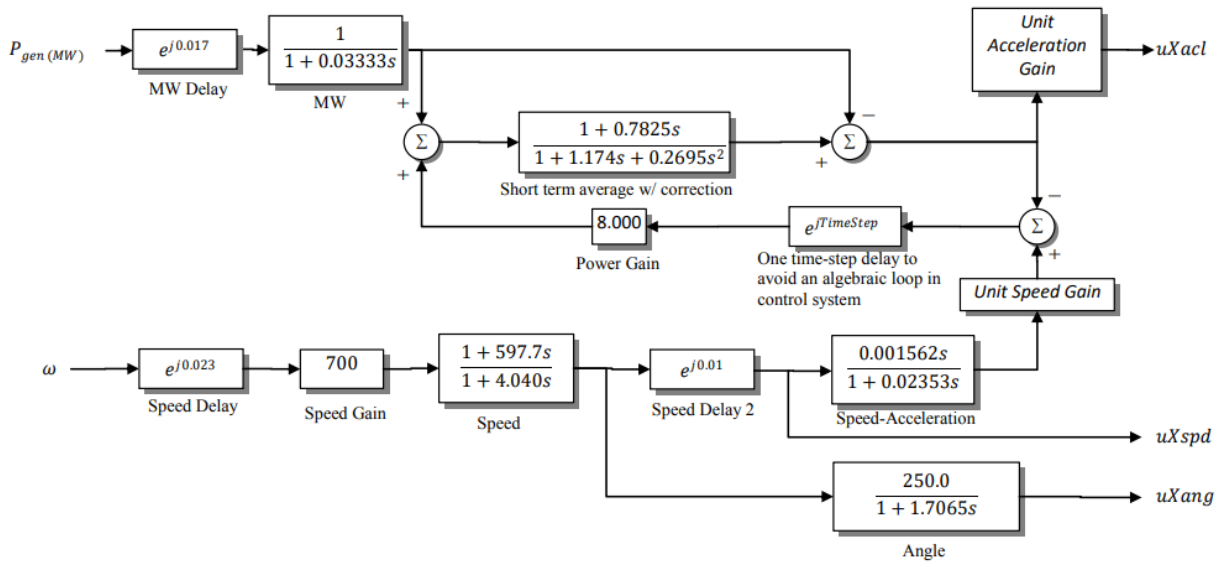

Figure 6. Individual Unit Calculations

This diagram represents calculations for one generating unit and thus would be applied to each Colstrip generator in simulation. Filters on the actual ATR are designed to remove torsional vibrations of the shaft. Because these dynamics are not typically modeled in transient simulations, the higher frequency filtering was removed for this model [7]. The delays shown on the inputs were added to model sensor delays.

The speed input in this model, ω, equals the per-unit (PU) speed deviation of a generator unit. PU speed is an expression of the speed quantity as a fraction of the base quantity. At 60-Hz nominal speed, this value would be 1. This input will be calculated by PST in simulation.

The electrical power input in this model, $P_{gen(MW)}$, represents the electric power output of a unit in megawatts. This value will also be calculated by PST in simulation.

The speed, angle, and acceleration outputs are labeled uXspd, uXang, and uXacl, respectively. X represents a number, 1 through 4, corresponding to the appropriate generator. According to Bahr, one unit of speed corresponds to 3.766E-3 radians per second, or one

thousandth of one percent of nominal speed, one thousand angle counts correspond to 1.44 degrees, and one unit of acceleration corresponds to 1 MW and 2.75 MW of acceleration power for a small and large unit, respectively [7].

The Simulink model that was provided for the unit calculations is shown in Figure 7. There are a few differences in this model from the one in Figure 6. First, different variable names are used for the inputs and outputs. Next, the speed gain was reduced from 700 to 660. Also, individual unit gains for power and speed are passed in as inputs instead of being hard-coded. Lastly, Mr. Bahr added an offset on the power input to initialize the transfer functions to zero.

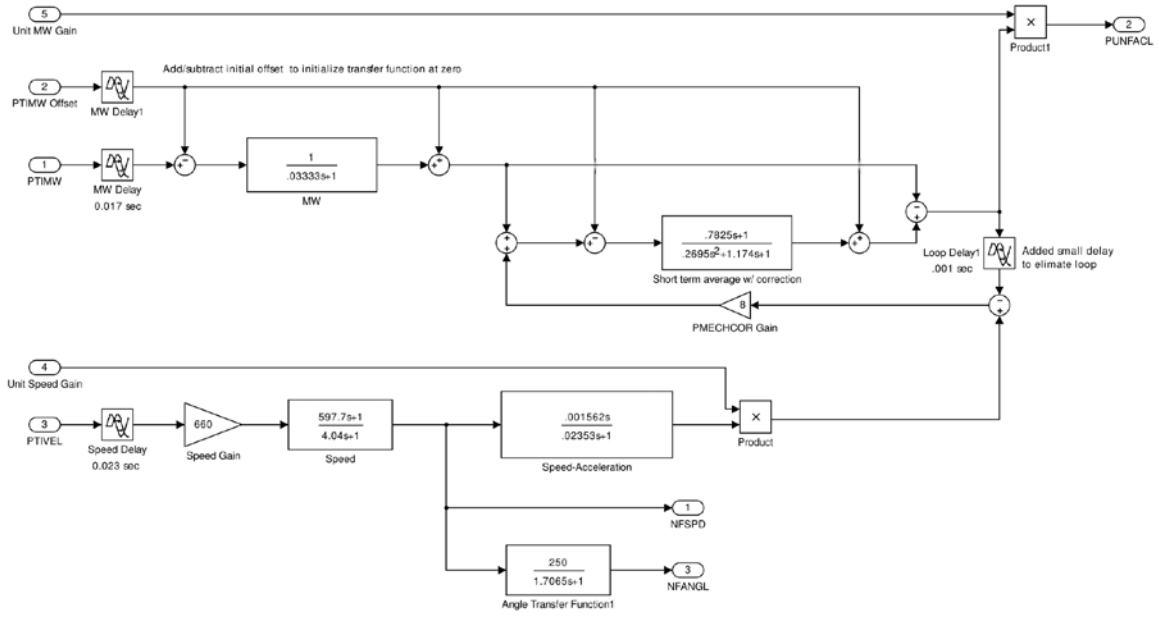

Figure 7. Simulink Unit Calculations Block Diagram

2.4.2. Simulink Center of Mass Calculations

The Simulink model of the center-of-mass calculations is shown in Figure 8. This block diagram has the purpose of applying equation (1). The online status is a 4-bit representation of the online units. This is deconstructed into individual unit statuses where they can be multiplied by their associated unit values to form the numerator of equation (1). The unit statuses are also

summed to form the denominator. A 2.75 gain is applied to the status and values of the two larger units.

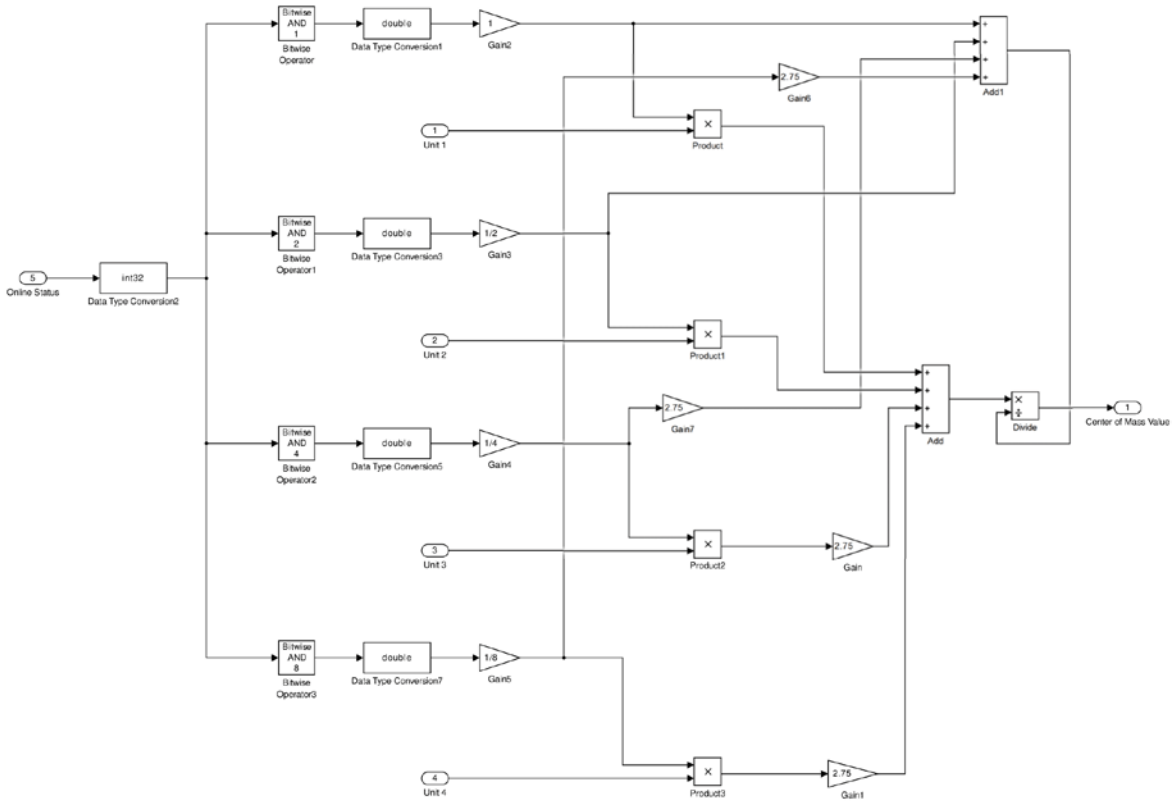

Figure 8. Simulink Center of Mass Calculations Block Diagram

2.4.3. Simulink Look-Ahead Angle Calculation

The Simulink model that performs the LA angle calculation is shown in Figure 9. This block diagram applies equation (2) by multiplying the speed input by 15, the acceleration by 35.971, and then summing them together with the angle.

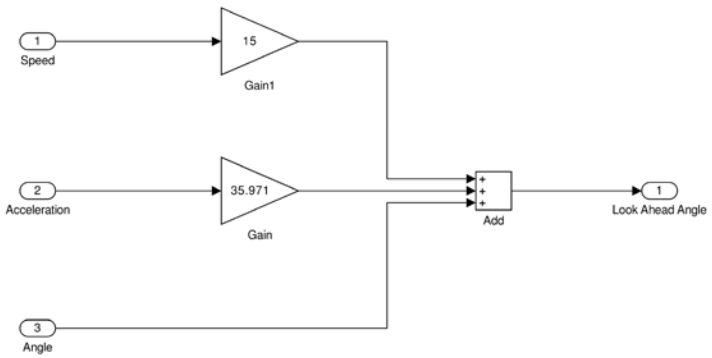

Figure 9. Simulink LA Angle Block Diagram

2.4.4. Simulink Trip Logic

The Simulink model for the trip logic and the individual trip algorithms is shown in Figure 10. The leftmost sub-block is the Gross Enable and the vertical sub-blocks are ten of the individual trip algorithms. These sub-blocks are composed of MATLAB code which can be found in Appendix I. The Gross Enable is used to enable or disable the individual algorithms by requiring a minimum amount of acceleration before setting an enable flag. This is used for protection against false trips. The output of each trip algorithm is a trip percent. These are passed into a block that determines the maximum trip percent value before being passed out of the block diagram.

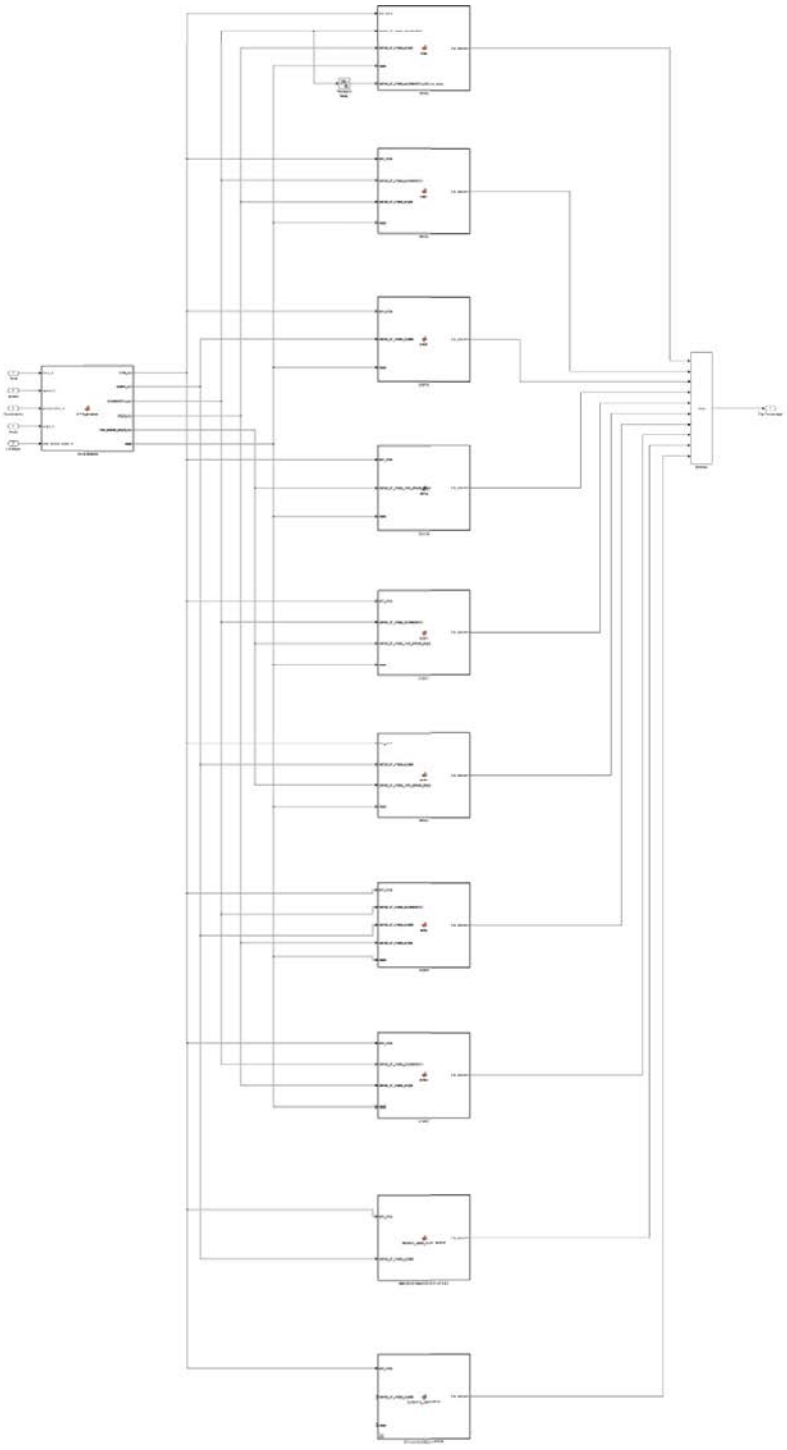

Figure 10. Simulink Trip Logic Block Diagram

2.5. ATR Implementation in PST

The Simulink model that was provided allows a user to conduct a PST simulation and then transfer generator speed and electric power data through the model post-simulation. This allows the user to check for correct ATR operation, but is generally cumbersome, requiring the user to perform two simulations for every case. But more importantly, the Simulink model does not allow for efficient closed-loop control; i.e. tripping generator units during simulation. This would require calling the Simulink model at each time-step of simulation. Opening and closing Simulink so often would be computationally inefficient and increase the total time to perform a simulation at least severalfold. Therefore, it was decided to modify the PST software to include the ATR algorithm as a function. This would allow PST simulations to run the ATR algorithm internally and trip individual units, if necessary, during the simulation. The ATR was implemented in PST by modifying a specialized generator trip logic function *mac_trip_logic*. This function was designed by Dr. Trudnowski to set a flag for a specific generator. If the flag was set, PST would then set the electric power of the generator to 0, effectively tripping it offline. This function would be modified to perform the ATR calculations and issue a trip flag to the appropriate Colstrip generator(s).

2.5.1. ATR Function Introduction

The final version of the modified trip function used to represent the ATR algorithm was named *mac_trip_logic_ATRV3* and the first section of code is shown in Figure 11. The complete function can be found in Appendix D. The inputs to the function are a vector containing the online status of all generators in the model, a storage matrix, a vector of simulation time, and an integer representing the current iteration of the simulation. The outputs of the function are a vector of desired trips and the storage matrix. The function begins by declaring electric power,

machine speed, and the number of machines as global variables to give the function access to this data. This is shown in lines 24-26.

```
1   function [tripOut,mac_trip_states] = mac_trip_logic(tripStatus,mac_trip_states,t,kT)
2   % Purpose: Model the ATR and trip generators.
3   %
4   % Inputs:
5   %   tripStatus = n_mac x 1 bool vector of current trip status.  If
6   %       tripStatus(n) is true, then the generator corresponding to the nth
7   %       row of mac_con is already tripped.  Else, it is false.
8   %   mac_trip_states = storage matrix defined by user.
9   %   t = vector of simulation time (sec.).
10  %   kT = current integer time (sample).  Corresponds to t(kT)
11  %
12  % Output:
13  %   tripOut = n_mac x 1 bool vector of desired trips.  If
14  %       tripOut(n)==1, then the generator corresponding to the nth
15  %       row of mac_con is will be tripped.  Note that each element of
16  %       tripOut must be either 0 or 1.
17  %   mac_trip_states = storage matrix defined by user.
18
19  % Version 1.3
20  % Authors:   Dan Trudnowski, RJ Hallett
21  % Date:   March 2017
22
23  %% Define global variables
24  global mac_spd %Gen pu speeds; mac_spd(n,kT) = gen n speed at time t(kT)
25  global pelect  %gen pu powers; pelect(n,kT) = gen n real power at time t(kT)
26  global n_mac
27
```

Figure 11. ATR Function Introduction

2.5.2. Initialization

The next section of the ATR function is shown in Figure 12 and covers the initialization sequence. First, the *tripOut* variable is created and set to false in line 29 to ensure no generators are initially tripped offline. Next an if-statement at line 31 ensures execution only during the first iteration of the simulation, when *kT* equals 1. Within the if-statement, the storage matrix *mac_trip_states* is initialized with 65 rows and a number of columns equal to the length of the simulation time vector. The first two rows of the first column of the storage matrix will store the

current index of iteration and current simulation time. These are initialized to zero in lines 33 and 34. In lines 36-58, the four Colstrip generators' initial speed and electric power are stored. Two rows were provided for each value of power and speed to allow for filtering. An additional row separates each generator's information to store a trip state. In lines 60 and 61, the online status of the Colstrip generators is initialized by converting binary '1111' to a decimal using the built-in MATLAB function *bin2dec*, and then storing the result in column 1; row 65 of the storage matrix. A 1 represents an online unit and a 0 represents an offline unit. During the initialization, all four units are assumed to be online, thus *onlineStatus* is set to 15, the decimal equivalent of '1111'.

```
28      %% Initialize
29 -    tripOut = false(n_mac,1);   %Set tripOut to zero's, length = # of mach's
30      % Initialize states
31 -    if kT==1
32 -        mac_trip_states = zeros(65,length(t)); %Initialize size of storage var
33 -        mac_trip_states(1,1) = 0; %time of last analysis (sample)
34 -        mac_trip_states(2,1) = 0; %time of last analysis (sec.)
35          %states for gen. 14 pelect filter
36 -        mac_trip_states(3,1) = pelect(14,1); %gen 14 initial real power (pu)
37 -        mac_trip_states(4,1) = pelect(14,1); %gen 14 initial real power (pu)
38          %states for gen. 35 pelect filter
39 -        mac_trip_states(8,1) = pelect(35,1); %gen 35 initial real power (pu)
40 -        mac_trip_states(9,1) = pelect(35,1); %gen 35 initial real power (pu)
41          %states for gen. 36 pelect filter
42 -        mac_trip_states(13,1) = pelect(36,1); %gen 36 initial real power (pu)
43 -        mac_trip_states(14,1) = pelect(36,1); %gen 36 initial real power (pu)
44          %states for gen. 37 pelect filter
45 -        mac_trip_states(18,1) = pelect(37,1); %gen 37 initial real power (pu)
46 -        mac_trip_states(19,1) = pelect(37,1); %gen 37 initial real power (pu)
47          %states for gen. 14 mac_spd filter
48 -        mac_trip_states(5,1) = mac_spd(14,1); %gen 14 initial speed (pu)
49 -        mac_trip_states(6,1) = mac_spd(14,1); %gen 14 initial speed (pu)
50          %states for gen. 35 mac_spd filter
51 -        mac_trip_states(10,1) = mac_spd(35,1); %gen 35 initial speed (pu)
52 -        mac_trip_states(11,1) = mac_spd(35,1); %gen 35 initial speed (pu)
53          %states for gen. 36 mac_spd filter
54 -        mac_trip_states(15,1) = mac_spd(36,1); %gen 36 initial speed (pu)
55 -        mac_trip_states(16,1) = mac_spd(36,1); %gen 36 initial speed (pu)
56          %states for gen. 37 mac_spd filter
57 -        mac_trip_states(20,1) = mac_spd(37,1); %gen 37 initial speed (pu)
58 -        mac_trip_states(21,1) = mac_spd(37,1); %gen 37 initial speed (pu)
59
60 -        onlineStatus = bin2dec('1111'); %Initialize online status
61 -        mac_trip_states(65,1) = onlineStatus;   %Save online status
62 -    end
63
```

Figure 12. ATR Function Initialization

2.5.3. Start Algorithm and Update Generator States

After the initialization is complete, the ATR algorithm can begin execution. Figure 13 shows how the generator speed and power states are updated at each iteration. On line 65, an if-statement ensures the ATR algorithm executes at a fixed rate of 60 sps. This is required because PST can potentially change the step size of the simulation and the discrete filters in the unit

calculations were derived to only work at 60 sps. Furthermore, 60 sps was chosen because this matches the expected sample rate of PMU measurements. After the if-statement, row 1; column 1 of the storage matrix is incremented in line 66. This number represents the number of iterations the algorithm has executed. The next line defines and updates a filter index, *k*, defined as one greater than the iteration number. Line 68 shows row 2; column 1 of the storage matrix updated with the current analysis time. Lines 71-84 update the storage matrix with the current speed and electric power of the generators from the data stored in the global variables. Line 87 updates the online status variable from the storage matrix.

```
64      %% Start ATR Algorithm and Update Generator States
65 -    if (t(kT)-mac_trip_states(2,1)) >= (1/60 - 1/600) %Ensure 60 sps sample rate
66 -        mac_trip_states(1,1) = mac_trip_states(1,1) + 1; %update analysis sample
67 -        k = mac_trip_states(1,1) + 1; %increment current filter index
68 -        mac_trip_states(2,1) = t(kT); %update analysis time
69
70          %Update gen 14 states
71 -        mac_trip_states(4,k) = pelect(14,kT);
72 -        mac_trip_states(6,k) = mac_spd(14,kT);
73
74          %Update gen 35 states
75 -        mac_trip_states(9,k) = pelect(35,kT);
76 -        mac_trip_states(11,k) = mac_spd(35,kT);
77
78          %Update gen 36 states
79 -        mac_trip_states(14,k) = pelect(36,kT);
80 -        mac_trip_states(16,k) = mac_spd(36,kT);
81
82          %Update gen 37 states
83 -        mac_trip_states(19,k) = pelect(37,kT);
84 -        mac_trip_states(21,k) = mac_spd(37,kT);
85
86          %Update online status
87 -        onlineStatus = mac_trip_states(65,1);
88
```

Figure 13. Update Gen States and Ensure 60 SPS Rate

2.5.4. ATR Calculations

The next section of the ATR function contains other functions to perform the calculations needed to detect events and initiate generator trips. These functions, shown in Figure 14, include the unit calculations, center-of-mass calculations, look-ahead angle calculation, trip logic, and trip arbitration. These functions are described in-depth below, but generally each section follows the same process; a calculation is performed by a function, the output is saved in the storage matrix, and is passed to the input of the next function. A filter index greater than 2 is required by the if-statement at line 90 because otherwise the 2-cycle delay found in the unit calculations function would result in an indexing error during the first iteration. Speed and power gains for the four generators are defined in lines 92 and 93 according to the Simulink model that was shown earlier. Corresponding to the actual ATR, unit calculations are performed first, on line 95.

```
89      %% Unit Calculations
90 -        if k>2
91              %Unit Gains
92 -            UnitSpeedGain = [12.5102, 12.5102, 34.4285, 34.4285];
93 -            UnitMWGain = [1,1,1/2.75, 1/2.75];
94              %Calculates speed, acceleration, angle
95 -            [NFSPD, PUNFACL, NFANGL, mac_trip_states] = unitCalc(mac_trip_states, UnitSpeedGain, UnitMWGain,k);
96      %% Center of Mass Calculations
97              %Averages speed, acceleration, and angle
98 -            speed = centerMass(NFSPD(1), NFSPD(2), NFSPD(3), NFSPD(4), onlineStatus);
99 -            accel = centerMass(PUNFACL(1), PUNFACL(2), PUNFACL(3), PUNFACL(4), onlineStatus);
100 -           angle = centerMass(NFANGL(1), NFANGL(2), NFANGL(3), NFANGL(4), onlineStatus);
101             %Save values in storage matrix
102 -           mac_trip_states(61,k) = speed;
103 -           mac_trip_states(62,k) = accel;
104 -           mac_trip_states(63,k) = angle;
105     %% Look-Ahead Angle Calculation
106 -           lookAheadAng = lookAhead(speed, accel, angle);  %Performs LA Angle calculation
107 -           mac_trip_states(64,k) = lookAheadAng;           %Save in storage matrix
108     %% Trip Logic
109             %Trip percentage saved in storage matrix
110 -           mac_trip_states = tripAlgorithms(t(kT), speed, accel, angle, lookAheadAng, mac_trip_states, k);
111     %% Trip Arbitration (for 48% trip)
112             %Comment out to run in open-loop
113 -           if mac_trip_states(60,k) == 0.48
114 -               tripOut([14, 36]) = true;     %Trip small and large gen
115 -               mac_trip_states([7,17],k) = 1;
116 -               mac_trip_states(65,1) = 10;   %Change online status
117 -           end
118 -       end
119 -   end
120 -   end
```

Figure 14. ATR Calculations, Trip Logic, and Trip Arbitration

2.5.4.1. Discrete-Time Filtering

The unit calculations function, *unitCalcs*, relies on discrete filters that were derived to replace the continuous-time filters found in the Simulink model. The derivation was accomplished using MATLAB's *c2d* function which converts continuous-time systems to discrete-time. The *c2d* function requires the coefficients of the continuous-time filter, the sampling time for the discrete-time filter, and a discretization method. The coefficients were taken from the Simulink model, the sampling time was set to 1/60, and the discretization method was selected as the 'Tustin' method which is a bilinear approximation. The code used to generate the discrete-time filter coefficients can be found in Appendix A. The accompanying bode plots can be found in Appendix B. A filtering function, *funFiltMacTrip*, was used repeatedly in the unit calculations to implement each discrete-time filter. This function requires the coefficients of the relevant discrete-time filter and the desired state values that are to be filtered. This function can be found in Appendix C.

2.5.4.2. Unit Calculations Function

The unit calculations function, shown in Figure 15, has inputs of the current filter index; *k*, the storage matrix; *mac_trip_states*, and respective speed and power gains; *UnitSpeedGain*, and *UnitMWGain*. This function has outputs of speed; *NFSPD*, angle; *NFANGL*, and acceleration; *PUNFACL*. These outputs are vectors, four values long, with a value corresponding to each unit.

```
1   function [NFSPD, PUNFACL, NFANGL, mac_trip_states] = unitCalc(mac_trip_states, UnitSpeedGain, UnitMWGain,k)
2   %Purpose: Perform Unit Calculations
3   %
4   %Inputs:
5   %       mac_trip_states = storage matrix
6   %       UnitSpeedGain = 1x4 vector containing unit speed gains
7   %       UnitMWGain = 1x4 vector containing unit power gains
8   %       k = filter index
9   %
10  %Outputs:
11  %       Speed, acceleration, and angle = 1x4 vectors containing values for
12  %           each unit
13  %       mac_trip_states = storage matrix
14  %
15  % Version 1.0
16  % Authors:  RJ Hallett
17  % Date:   March 2017
18  load filters.mat
19  SpeedG = 660;
20  PmechG = 8;
21  %R - Acceleration from last iteration
22  R = mac_trip_states(47:50,k-1);
23  PTIMWoffset = mac_trip_states(4:5:19,1);    %Power Offset
24  %P1 - Power deviation signal
25  mac_trip_states([26:29],k) = mac_trip_states([4:5:19],k-1)-PTIMWoffset;
26  %S1 Speed Deviation*Speed Gain
27  mac_trip_states([23, 24, 25, 26],k) = (mac_trip_states([6, 11, 16, 21],k-2)-1)*SpeedG;
28  for i = 1:4 %For each generator
29      %P2 - Filter 1 output
30      mac_trip_states = funFiltMacTrip(G1.num{1},G1.den{1},mac_trip_states, 26+i, 30+i,k);
31      %P3 - Add back power offset
32      P3(i) = mac_trip_states(30+i,k)+PTIMWoffset(i);
33      %S2 - Filter 2 output
34      mac_trip_states = funFiltMacTrip(G3.num{1},G3.den{1},mac_trip_states, 22+i, 34+i,k);
35      NFSPD(i) = mac_trip_states(34+i,k);     %Speed output
36      %Filter 3 output
37      mac_trip_states = funFiltMacTrip(G5.num{1},G5.den{1},mac_trip_states, 34+i, 38+i,k);
38      NFANGL(i) = mac_trip_states(38+i,k);    %Angle output
39      %A1 - Filter 4 output
40      mac_trip_states = funFiltMacTrip(G4.num{1},G4.den{1},mac_trip_states, 34+i, 42+i,k);
41      %A2 - Multiply A1 by speed gains, subtract by previous Accel value, multiply by mech power gain
42      A2(i) = (mac_trip_states(42+i,k)*UnitSpeedGain(i)-R(i))*PmechG;
43      %A3 - Add mech accel to electric accel, subtract by power offset
44      mac_trip_states(50+i,k) = A2(i) + P3(i) - PTIMWoffset(i);
45      %A4 - Filter 5 output
46      mac_trip_states = funFiltMacTrip(G2.num{1}, G2.den{1}, mac_trip_states, 50+i,54+i,k);
47      %A5 - Add back offset, subtract by P3
48      A5(i) = mac_trip_states(54+i,k)+PTIMWoffset(i) - P3(i);
49      mac_trip_states(46+i,k) = A5(i);        %Save A5 for next itereation
50      PUNFACL(i) = UnitMWGain(i)*A5(i);       %Multiply by MW gains, set to Accel output
51  end
```

Figure 15. Unit Calculations Function

The unit calculations function starts with loading the discretized filter coefficients, defining the speed gain as 660, and the mechanical power gain as 8, which is shown in lines 18-20 of Figure 15. In line 22, a temporary variable, *R*, is set equal to the acceleration value from the previous iteration. This variable is necessary for the feedback loop and its location relative to the Simulink model is shown in Figure 16. In line 23, an electric power offset is created and set

equal to the initial electric power of each generator from the storage matrix. In line 25, this is subtracted from the electric power, one iteration delayed, to initialize the inputs for the first filter to 0, and to account for the delay on the power input in the Simulink model**.** This creates the power deviation signal, P1, which is labeled in Figure 16. Unit speed deviation is calculated next in line 27 by subtracting the PU speed, two iterations delayed, by 1. This is multiplied by the speed gain to create the signal S1. The two-iteration delay accounts for the delay on the speed input found in the Simulink model. A for-loop at line 28 is used to repeat the remaining calculations for each generator. At line 30 the P1 signal passes through the filtering function to create signal P2. The offset is added back in line 32 to make signal P3. In line 34 the speed deviation S1 passes through the filtering function to create signal S2. This is set equal to the speed output *NFSPD*. S2 passes through the filtering function in line 37 to create the angle output *NFANGL*. In line 40, S2 passes through the filtering function to produce the mechanical acceleration signal A1. This is multiplied by the speed gain, then subtracted by the feedback acceleration value, R, and finally multiplied by the mechanical power gain to create signal A2 in line 42. This is added to P3 and subtracted by the power offset on line 44 to create signal A3. In line 46 this signal is passed through the filtering function to generate signal A4. The power offset is added back, and the result is subtracted by P3 to create signal A5 in line 48. This is saved in the storage matrix to calculate R for the next iteration. A5 is multiplied by the unit power gain in line 50 to generate the last output *PUNFACL*.

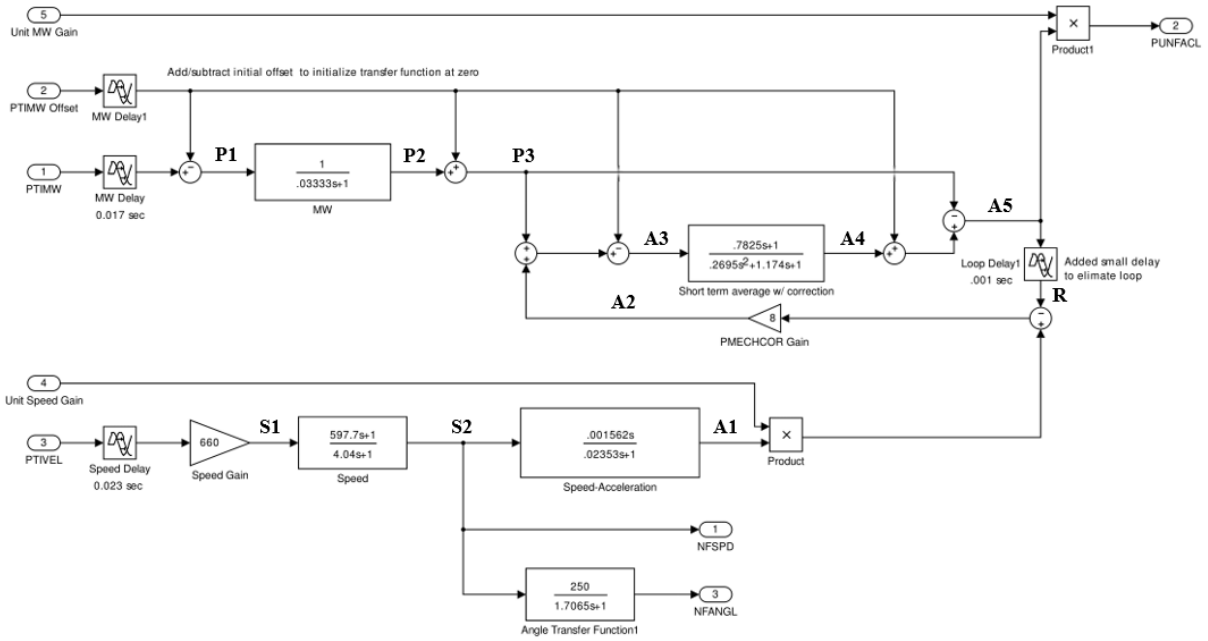

Figure 16. Unit Calculations with Labeled Signals

2.5.4.3. Center of Mass Function

The center-of-mass function, *centerMass*, is shown in Figure 17 and is called three times in lines 98-100 of Figure 14; once for each output of the *unitCalc* function: *NFSPD*, *PUFACL*, *NFANGL* (speed, acceleration, and angle). The center-of-mass function has an input for each unit's respective value and the *onlineStatus* variable.

```
1   function [cmVal] = centerMass(u1val, u2val, u3val, u4val, onlineStatus)
2   %Purpose: Calculate the center of mass average
3   %
4   %Inputs:
5   %       uXval = Speed, Accel, or Angle value corresponding to the unit X,
6   %           where X is a number from 1 - 4
7   %       onlineStatus = decimal representation of online status
8   %Outputs:
9   %       cmVal = center of mass value
10  %
11  % Version 1.0
12  % Author:   RJ Hallett
13  % Date:   March 2017
14  %
15  %Calculate inidvidual unit status:
16  u1stat = double(bitand(1, onlineStatus));       %bitwise AND to calculate gen 1 status
17  u2stat = (1/2)*double(bitand(2, onlineStatus)); %bitwise AND to calculate gen 2 status
18  u3stat = (1/4)*double(bitand(4, onlineStatus)); %bitwise AND to calculate gen 3 status
19  u4stat = (1/8)*double(bitand(8, onlineStatus)); %bitwise AND to calculate gen 4 status
20
21  %Center of mass calculation
22  cmVal = (u1stat*u1val + u2stat*u2val + 2.75*(u3stat*u3val + u4stat*u4val))/(u1stat + u2stat + 2.75*(u3stat+u4stat));
23  end
```

Figure 17. Center of Mass Calculation Function

The first step in this function is to calculate the status for each unit which is done in lines 16-19 of Figure 16. The status for unit 1, *u1stat*, is calculated by bitwise ANDing 1 (binary 0001) with *onlineStatus*. 1 is returned if the generator is online and a 0 is returned if offline. This corresponds directly to the method used in the Simulink model shown in Figure 18 with signals labeled. The status for unit 2, *u2stat*, is calculated by ANDing 2 (binary 0010) with *onlineStatus*. The return is 2 if online so the result is divided by 2 to restrict the status to either 1 or 0. For *u3stat* this process is repeated by ANDing *onlineStatus* with 4 (binary 0100), and for *u4stat* by ANDing *onlineStatus* with 8 (binary 1000). The returns are divided by 4 and 8 respectively. With the individual status values calculated, Equation 1 is applied on line 22 by multiplying the respective unit values by their unit status and then dividing by the sum of the status values. A gain of 2.75 is applied to the values and status of the two larger units. The outputs are saved in the storage matrix in lines 102-104 of Figure 14.

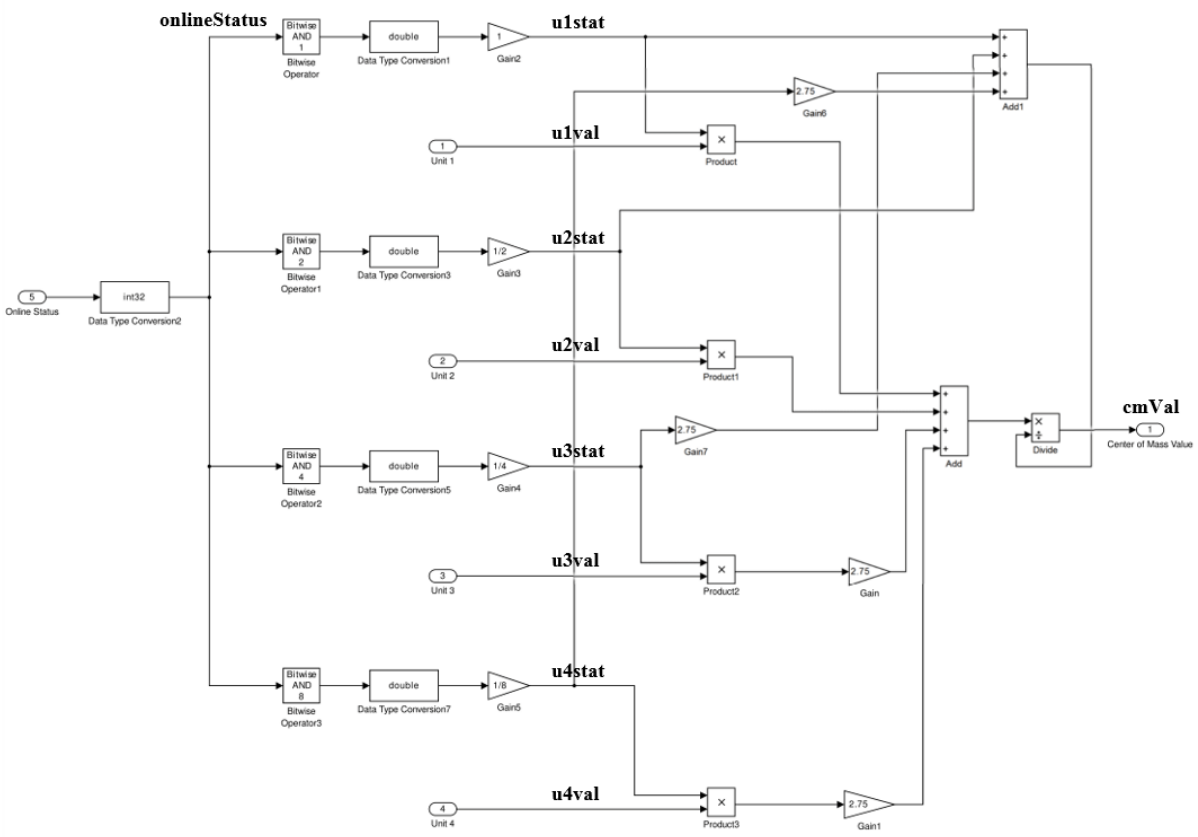

Figure 18. Center of Mass Calculation with Labeled Signals

2.5.4.4. Look-Ahead Angle Function

The LA angle is calculated on line 106 of Figure 14 using the function *lookAhead*. This function is shown in Figure 19 and the inputs are the outputs of the center of mass function: *speed*, *angle*, and *accel*. Equation (2) is used in line 17 to calculate the LA angle. The result is passed out as the variable *lookAheadAng* and saved in the storage matrix on line 107 of Figure 14.

```
1   function [lookAheadAng] = lookAhead(speed, accel, angle)
2   %Purpose: Calculate look-ahead angle
3   %
4   %Inputs:
5   %       Speed = average speed
6   %       Accel = average acceleration
7   %       Angle = average angle
8   %
9   %Outputs:
10  %       lookAheadAng = look-ahead angle
11  %
12  % Version 1.0
13  % Author:   RJ Hallett
14  % Date:   March 2017
15  %
16  %LA angle calculation:
17  lookAheadAng = speed*15+accel*35.971+angle;
18  end
```

Figure 19. Look-Ahead Angle Calculation Function

2.5.4.5. Trip Logic Function

Next, the function, *tripAlgorithms*, is called on line 110 of Figure 14 to detect events and issue a trip percentage. The code for this function is shown in Figure 20. The inputs are the variables *speed*, *angle*, *accel*, *lookAheadAng*, the storage matrix, the filter index, and the current time. At line 16 and 17 the *gross_enable* function is called to enable or disable a reset flag which is used to arm the individual trip functions. The code for *gross_enable* was copied directly from the Simulink model. Next an acceleration value, 4-cycles delayed, is created at line 23 to be used as an input for the first trip algorithm on line 29. An if-statement at line 22 prevents an indexing error if the filter index is 4 or less. From lines 29-47, 10 individual trip algorithms are called as functions. Each can issue its own trip percentage. The code for the trip algorithms was copied directly from the Simulink model and when creating the functions. The maximum trip percentage from the algorithms is saved in the storage matrix on line 49 and is passed out of the function.

```
function [mac_trip_states] = tripAlgorithms(time_in, speed_in, acceleration_in, angle_in, look_ahead_angle_in, mac_trip_states, k)
%Purpose: Detect events and issue trip percentage
%
%Inputs:
%       time_in = simulation time, angle_in = center of mass angle,
%       look-ahead angle = center of mass look-ahead angle,
%       mac_trip_states = storage matrix, k = filter index
%
%Output:
%       mac_trip_states = storage matrix with trip percentage placed in row 60
% Version 1.0
% Author:   RJ Hallett
% Date:   March 2017
%
%% Gross Enable
[time_out, speed_out, acceleration_out, angle_out, look_ahead_angle_out, reset]=...
                    gross_enable(time_in, speed_in, acceleration_in, angle_in, look_ahead_angle_in);
%Rename variables
sim_time = time_out; center_of_mass_speed = speed_out; mac_trip_states(59,k) = acceleration_out;
center_of_mass_acceleration = acceleration_out; center_of_mass_angle = angle_out;
center_of_mass_look_ahead_angle = look_ahead_angle_out;
if k>4
    center_of_mass_acceleration_4cycle_delay = mac_trip_states(59, k-4);    %Creates 4 cycle delayed acceleration value
else
    center_of_mass_acceleration_4cycle_delay = acceleration_out;
end
%% Trip Algorithms
%TDAC (Time Triggered Acceleration)
tripPrct1 = tdac(sim_time, center_of_mass_acceleration, center_of_mass_angle, reset, center_of_mass_acceleration_4cycle_delay);
%OACL (Over-Acceleration)
tripPrct2 = oacl(sim_time, center_of_mass_acceleration, center_of_mass_angle, reset);
%OSPD (Over-Speed)
tripPrct3 = ospd(sim_time, center_of_mass_speed, reset);
%OANG (Over-Angle)
tripPrct4 = oang(sim_time, center_of_mass_look_ahead_angle, reset);
%ACAN (Acceleration vs LA Angle)
tripPrct5 = acan(sim_time, center_of_mass_acceleration, center_of_mass_look_ahead_angle, reset);
%SPAN (Speed vs LA Angle)
tripPrct6 = span(sim_time, center_of_mass_speed, center_of_mass_look_ahead_angle, reset);
%ACSP (Acceleration vs Speed)
tripPrct7 = acsp(sim_time, center_of_mass_acceleration, center_of_mass_speed, center_of_mass_angle, reset);
%ANAC (Angle Triggered Acceleration)
tripPrct8 = anac(sim_time, center_of_mass_acceleration, center_of_mass_angle, reset);
%Second Peak Over-Speed
tripPrct9 = second_peak_over_speed(sim_time, center_of_mass_speed);
%Dynamic Oscillation
tripPrct10 = dynamic_oscillation(sim_time, center_of_mass_speed, reset);
%% Save maximum trip percentage
mac_trip_states(60,k) = max([tripPrct1, tripPrct2, tripPrct3, tripPrct4, tripPrct5, tripPrct6, tripPrct7, tripPrct8, tripPrct9, tripPrct10]);
end
```

Figure 20. Trip Algorithms Function

2.5.5. Trip Arbitration

With the trip percentage calculated, a trip signal can be issued to the individual generators. There are six levels of trip percentage: 14% (one small unit), 31% (one large unit), 48% (one small and one large), 64% (two large units), 81% (two large and one small), and 100% (all four units). The amount of generation tripped is greater than or equal to the actual trip call percentage. The trip arbitration, which decides which generators to trip off to satisfy the trip percentage, was not included in the Simulink model so a simplified form was added for closed-loop control in lines 113-117 of Figure 14. These lines are shown again in Figure 21. This trip arbitration only applies to a 48% trip, but the same format can be applied to the other possible

trip percentages. An if-statement at line 113 checks row 60 for a trip of 0.48. If this is true, the next line sets a flag to true for Colstrip units 1 and 3, which correspond to generators 14 and 36 in the modified MiniWECC. In line 115, the trip decision is also stored in the row and column of the storage matrix corresponding to the appropriate unit and time step. The next line changes row 65; column 1 of the storage matrix to the binary equivalent of '0101' which will update the variable *onlineStatus* in the next iteration to account for units 1 and 3 tripping offline.

```
111     %% Trip Arbitration (for 48% trip)
112             %Comment out to run in open-loop
113 -           if mac_trip_states(60,k) == 0.48
114 -               tripOut([14, 36]) = true;     %Trip small and large gen
115 -               mac_trip_states([7,17],k) = 1;
116 -               mac_trip_states(65,1) = bin2dec('0101');   %Change online status
117 -           end
```

Figure 21. Trip Arbitration Code

2.6. Validating PST ATR Model

To validate that the PST-integrated version of the ATR functioned the same as the Simulink version, the PST model was first placed in open-loop mode by commenting out the trip arbitration. This was done because the Simulink ATR model would not be operated in a closed-loop mode. A transient unstable event was simulated by faulting the line between buses 83 and 84 on the MiniWECC diagram, shown in Figure 22, for 11 cycles.

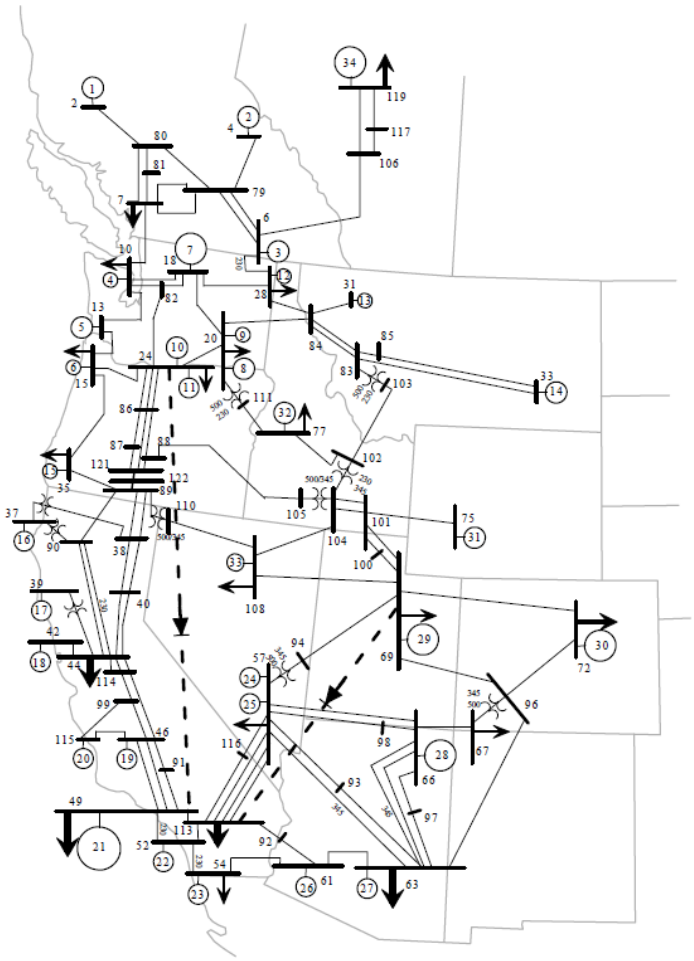

Figure 22. MiniWECC One-Line Diagram

After the PST simulation was performed, the simulation data was passed into the Simulink ATR model. The center-of-mass values and trip decisions were compared between the Simulink model and the PST-integrated model. Center-of-mass speed, angle, acceleration, and look-ahead angle calculated by each model are shown in Figure 23. Note that the dotted and solid lines from each model lie on top of each other. These plots verify that the two models are nearly identical. Small differences were the result of approximation errors created when the Simulink filters were discretized.

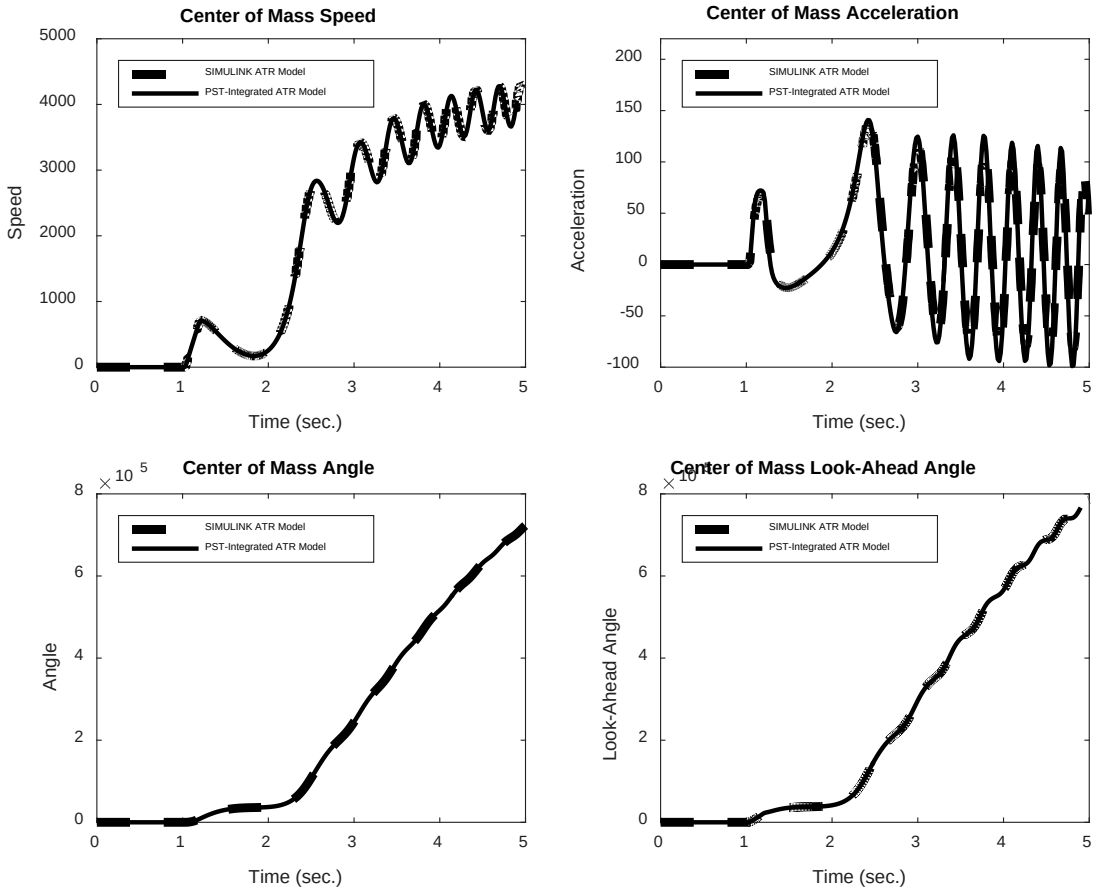

Figure 23. Center of Mass Comparisons

Table I below shows the trip decision that would have been issued by each ATR model. For this event, both models issue a 48% trip from the same trip algorithm. This trip percentage is equal to one large and one small unit being tripped. The PST-integrated ATR model trips 3.3 ms after the Simulink version, which is about 20% of a single 60 Hz power system cycle. This was expected, however, because the two models operate at different sampling speeds. The Simulink ATR model operates at 300 sps while the PST-integrated model is restricted to 60 sps. The time step of the Simulink model simulation is therefore 3.3 ms and thus explains the difference in trip times.

Table I: Trip Decision

	Trip Time (s)	Trip Percent	Trip Algorithm
Simulink	1.1466	48%	ACSP (Acceleration vs Speed)
PST	1.1500	48%	ACSP (Acceleration vs Speed)

2.7. ATR Closed-Loop Simulation

The next simulation involved the same 11-cycle fault, except the ATR's trip arbitration was restored in PST so it could be operated in closed-loop mode. The results from this simulation are provided to demonstrate the ATR stabilizing Colstrip by removing a percentage of the total generation during what would have been a transient unstable event. The center-of-mass values are shown in Figure 24 and can be compared to Figure 23 to illustrate the difference between closed-loop and open-loop operation. Note that the center-of-mass values from the closed-loop simulation indicate that Colstrip stabilizes after the trip is issued.

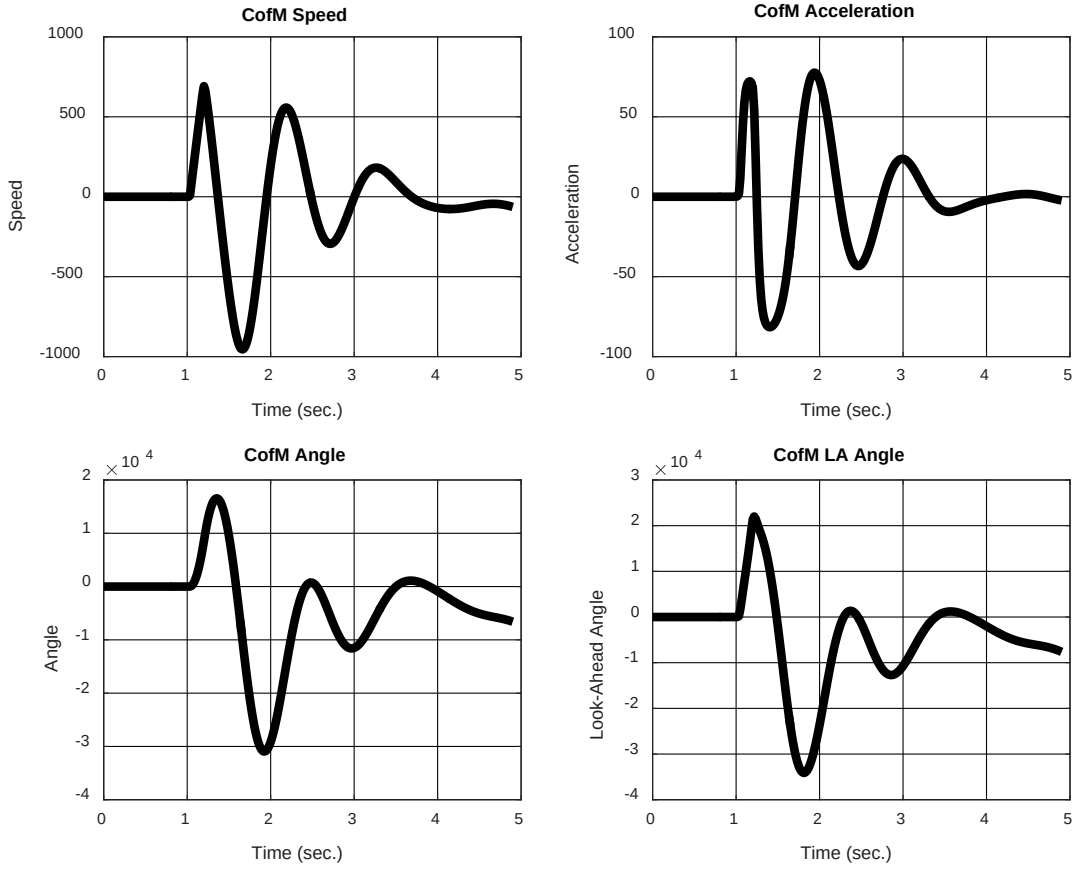

Figure 24. Center of Mass Closed-Loop Values

Another way to show the stabilizing effect of the ATR is to compare the electric power output of the four units in both open-loop and closed-loop cases. The individual electric powers in open-loop control are shown in Figure 25 while the individual electric powers in closed-loop control are shown in Figure 26. Note that the generators go transient unstable in open-loop control but remain synchronized in closed-loop control after two units are tripped offline.

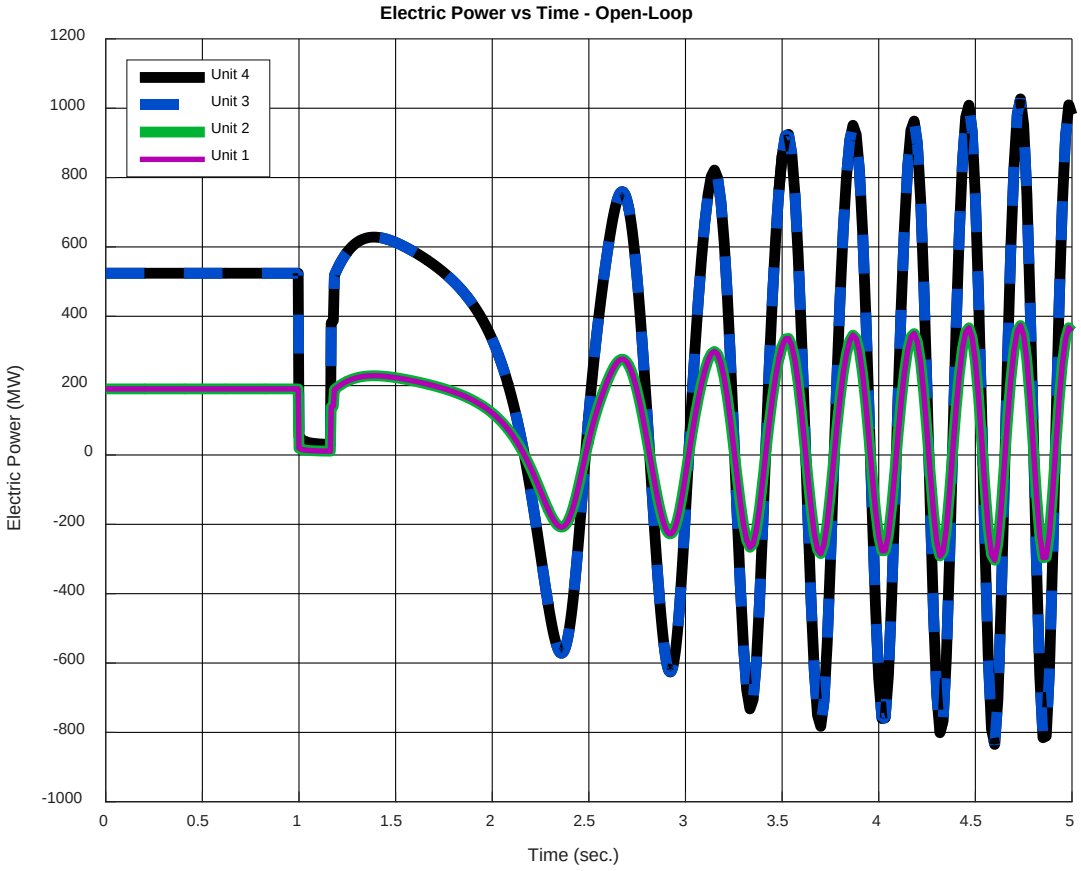

Figure 25. Colstrip's Electric Power in Open-Loop

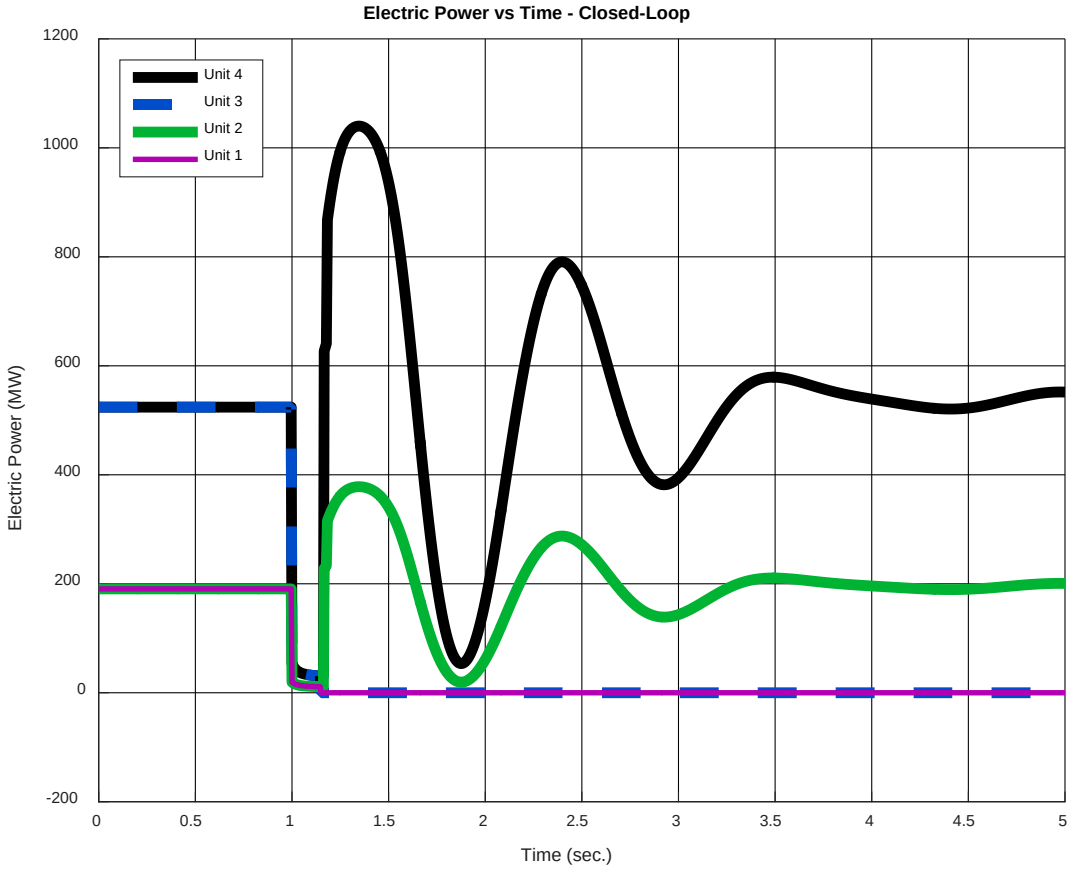

Figure 26. Colstrip's Electric Power in Closed-Loop

3. Synchrophasor-Based Control Scheme

3.1. Considerations

There are several problems that must be considered when developing a control scheme based on wide-area synchrophasor measurements. One of the key design issues is the accuracy of the PMU signal. A PMU measures voltage magnitude and angle, both of which are susceptible to discontinuities during switching events in the electrical network. Additionally, a contingency such as a fault near a PMU can cause large distortions to, or the complete loss of the voltage measurement. Ideally, PMU measurements must be accurate even during switching events and contingencies. This can be achieved either through filtering or state estimation, or the control scheme must be able to recognize inaccurate measurements and act appropriately.

Another design issue is the reliability of the PMU communication network. Long-distance communications will inevitably have higher rates of failure as distance increases. A control scheme based on long-distance communications must therefore have the ability to recognize failures and act appropriately.

The last issue is the speed of the PMU measurements. Ideally, measurements would be available instantaneously, but the reality is that even small delays produced by the PMU device combined with communication delays could have large consequences on the overall function of a control scheme. Delays several power system cycles in length could be the difference between reacting appropriately and exasperating an event. Fortunately, as PMU and fiber technology advance, their associated time delays continue to shrink.

Above all, the philosophy of ensuring “no harm” must be followed in the development of the synchrophasor-based control scheme. The idea is to not only account for accuracy, reliability, and speed, but also to ensure that the proposed control scheme does not interfere with the normal

operations of the existing ATR. The goal becomes to improve the selectivity of the ATR without decreasing the reliability.

3.2. Proposed Arming Scheme

The scope of this project was to study the feasibility of using wide-area measurements, therefore the extent of each design issue surrounding PMU measurements was not fully investigated. For the purposes of this research, it was assumed that PMU measurements would be perfectly accurate, reliable, and available instantaneously. However, the key requirement remains that if the PMU network fails to provide information, the ATR will operate in its traditional form.

The proposed control scheme is fairly simple and is shown below in Figure 27. The control signal, Δf, represents the relative frequency between the generation bus and a reference bus, measured by a PMU network, as shown in (3):

$$\Delta f = f_{local} - f_{ref} \tag{3}$$

Relative frequency is proposed for the control scheme because it is a robust indicator of the transient condition and can also be provided by a PMU. If the difference in frequency is equal to or below a threshold:

$$|\Delta f| \leq \Delta f_{max} \tag{4}$$

The ATR is disarmed and prevented from issuing a trip. If the difference in frequency is above the threshold:

$$|\Delta f| > \Delta f_{max} \tag{5}$$

The ATR is armed and allowed to trip. Conveniently, there is already a mechanism that arms and disarms the trip algorithms, the Gross Enable, which requires a minimum threshold of

acceleration to be reached in order to enable the individual trip algorithms. The Gross Enable could be easily modified to also require Δf to pass the threshold Δf_{max} before arming the trip algorithms. The max allowed relative frequency threshold would require tuning through extensive transient stability studies similar to how other ATR settings were found.

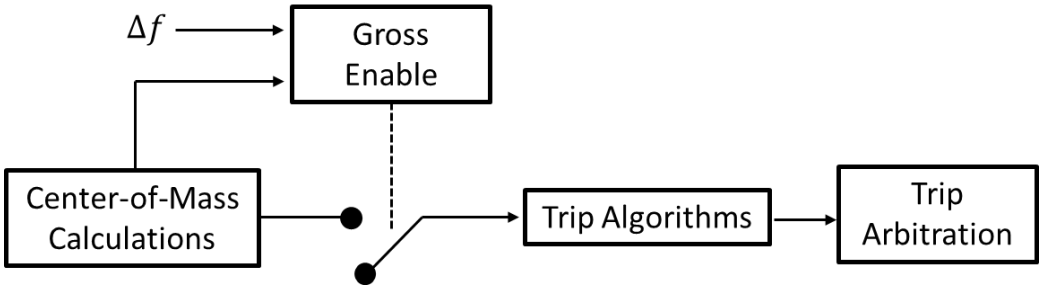

Figure 27. ATR Arming. The Gross Enable is augmented with relative frequency.

4. ATR Simulations

4.1. Case 1: Local Fault

The next step in this research was to determine if the proposed control scheme can improve the trip decisions of the ATR. A simulation was performed in the MiniWECC by faulting the line between bus 83 and 84 at t=1.0s, and then clearing the fault after 11 cycles by opening the line. The resulting simulation is shown in Figure 28. Local frequency deviation from the Colstrip bus (bus 33) is shown as the black trace. Remote frequency deviation was provided from the Grand Coulee bus (bus 18) as the blue trace. This bus was chosen because measurements from this location would provide a good indication of the frequency of the bulk power system. Relative frequency Δf is shown as the green trace.

The red vertical line near t=1.15s represents the time at which the ATR signals for a trip. Note that the units were not actually tripped as the point of these simulations was to determine if a trip was necessary. Because this fault was local to the Montana transmission system, the effect it has on Grand Coulee is minimal even though Colstrip becomes unstable. The relative frequency signal indicates that Colstrip is losing synchronism with the system, therefore the ATR made a correct trip decision. The horizontal red line represents a possible threshold level Δf_{max} for the proposed arming scheme. This value is 0.005 pu (or 3 Hz) and since the green trace is above this threshold at the time of the trip, the ATR would have been armed.

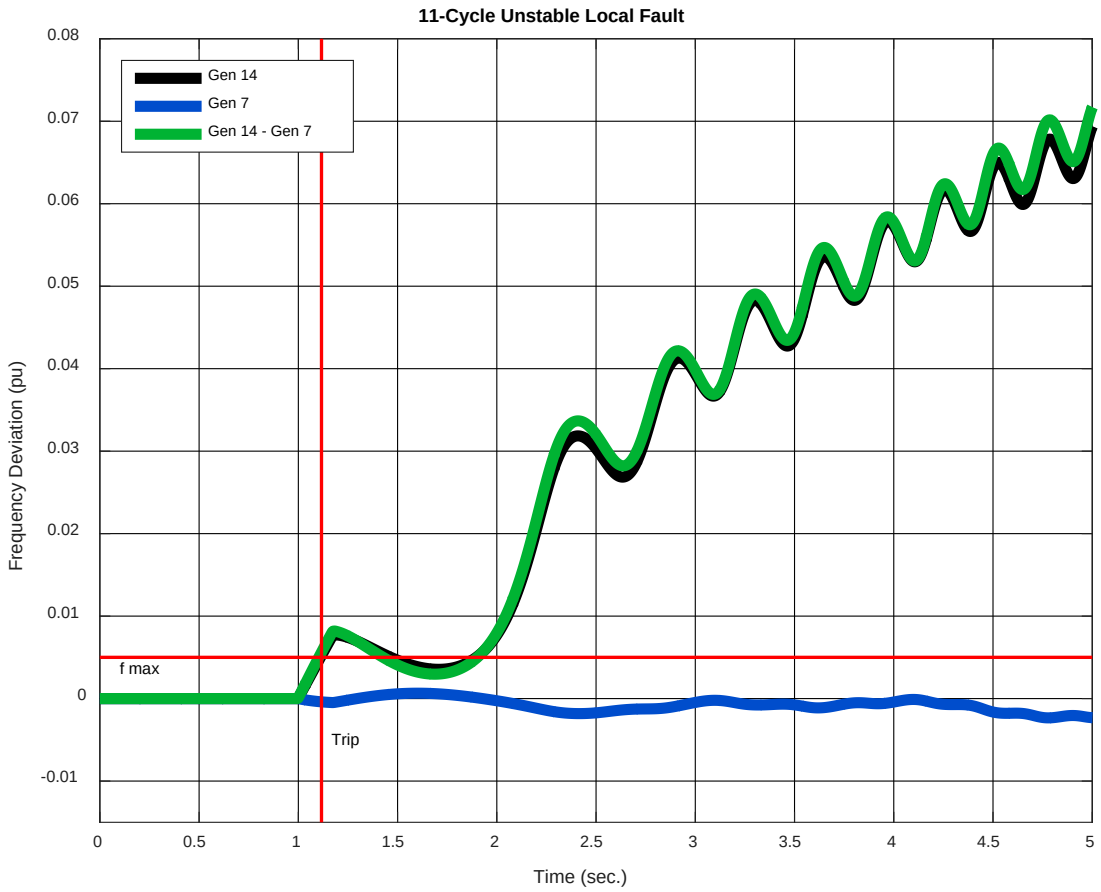

Figure 28. Colstrip, Grand Coulee, and Relative Frequency Response for Unstable Local Fault

4.2. Case 2: Remote Generator Trip

The next case that was studied involved a generator trip outside of the Montana transmission system. Many cases were looked at but only one generator trip was found to cause the ATR to trip. This was the loss of Gen 21 at t=1.0s which is shown in Figure 29. Again, the frequency deviation of Colstrip is shown as the black trace, Grand Coulee as the blue trace, and their relative frequency as the green trace. The frequency separation indicates that Colstrip begins to lose synchronism from Grand Coulee and therefore the ATR issued a correct trip decision as indicated by the red vertical line at t=4.1s. The units are not actually tripped as the

goal was to determine if a trip was necessary. Relative frequency indicates the instability to a greater extent than the local information as reflected by the larger magnitude of the green trace. If Δf_{max} was set to 0.005 Hz as indicated by the horizontal line, the ATR would have been armed and allowed to trip normally. The results of this simulation provide evidence that the proposed arming scheme would not interfere with the existing ATR algorithm for a remote disturbance that causes transient instability.

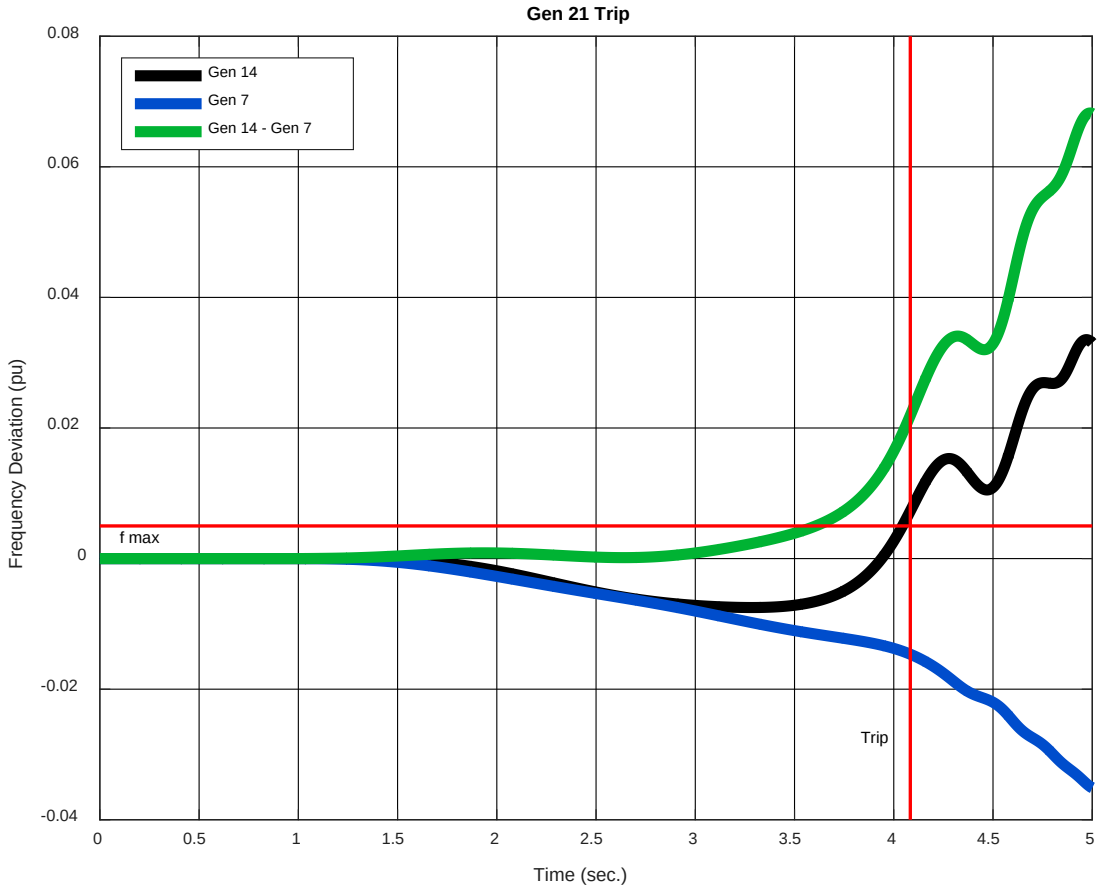

Figure 29. Colstrip, Grand Coulee, and Relative Frequency Response for Gen 21 Trip

4.3. Case 3: Remote Load Shed

The goal, however, was to show that the proposed control scheme could prevent a false trip. Ideally, this would be an event where the ATR issues a trip, but relative frequency indicates Colstrip is not separating from the system. To find an event where the ATR might false trip, many cases were studied simulating remote faults, or faults outside of the Montana transmission system. It was found that the ATR never false tripped for any of these cases.

It was concluded that a remote fault alone would be insufficient to cause the ATR to false trip. It was decided that a fault followed by a load-shed might cause a disturbance significant enough to false trip the ATR. Therefore, a 10-cycle fault was simulated between bus 7 and 8 in the MiniWECC at t=1.0s and then the line was opened. This was followed by a 1500MW load trip on bus 20 at t=1.4s, and at t=1.8s, an additional 4000MW load trip on bus 24. This represents a severe contingency known as a cascading failure. These types of failures are not uncommon in power grids because a single failure will often overload nearby systems resulting in a chain reaction. The results from the simulation are shown in Figure 30. The local frequency deviation of Colstrip is shown as the black trace, Grand Coulee as the blue trace, and relative frequency as the green trace. The traces show that Colstrip and the rest of the system accelerate together due to the load trip. Although there is some oscillation between the two generators, it is clear that they remain synchronized, i.e., Colstrip is transient stable relative to the generators in the Pacific Northwest. The ATR, however, calls for a trip at t=3.0s as indicated by the red vertical line. This trip is not desired and should be considered a false trip. If the proposed arming scheme was employed, relative frequency would have been below the 0.005 pu threshold as indicated by the horizontal red line, and the ATR would have been disarmed.

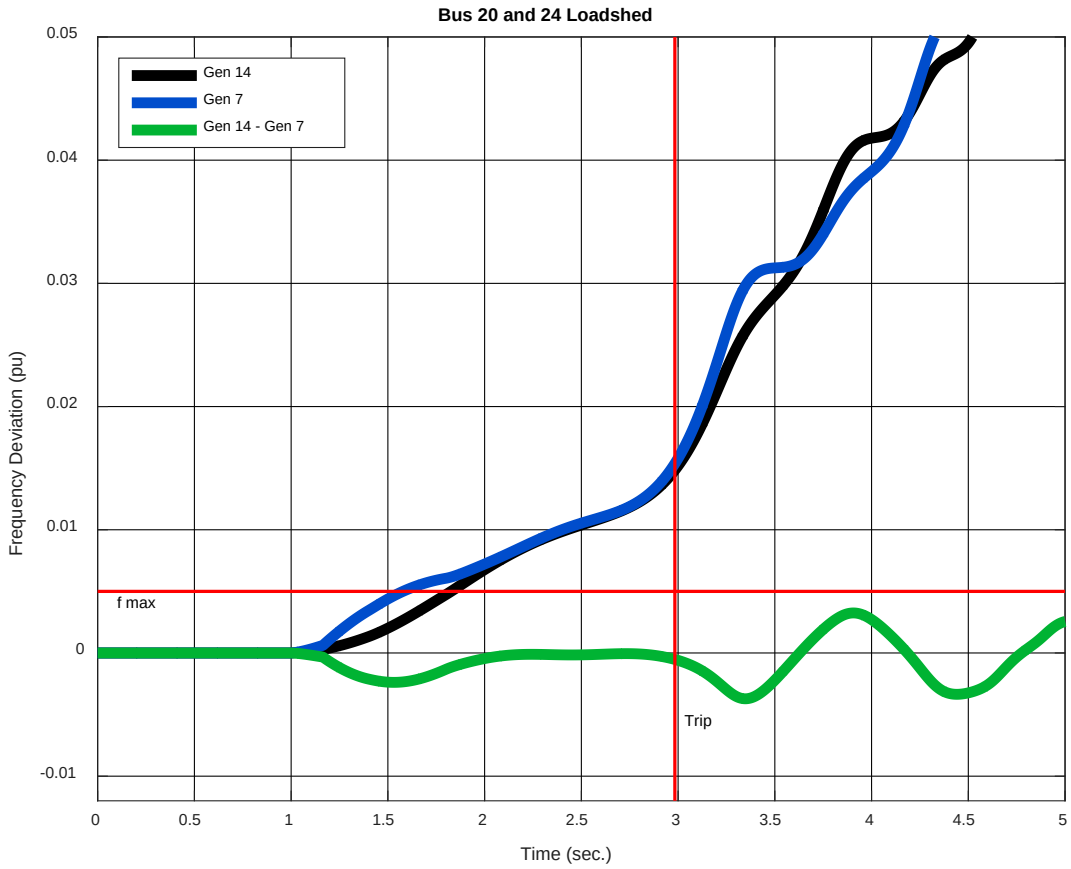

Figure 30. Colstrip, Grand Coulee, and Relative Frequency Response for Remote Load Shed

4.4. Conclusion

This research studied the benefits and feasibility of augmenting wide-area synchrophasor measurements with the existing ATR algorithm. The proposed arming control scheme would disarm the ATR if the relative frequency measured by a PMU network is below a predefined threshold. Two simulations demonstrated that this arming scheme would not harm the existing algorithm and one simulation showed that it could increase the selectivity of the ATR by preventing a false trip.

This research expanded on the work done in [5] in several ways. First, this research proposed a method of implementing PMU measurements by augmenting the existing ATR

algorithm with an arming scheme, as well as suggesting measuring relative frequency for the control signal as an alternative to measuring relative phase angle. Also, this research studied a variety of cases to evaluate the proposed arming scheme, where [5] only studied a single historical case.

This research was based on several major assumptions. Relative frequency measurements were assumed to be perfectly accurate, reliable, and instantaneously available. In reality, PMU measurements will be inaccurate during switching events, potentially suffer from communication failures, and inherently have some delay. Additional supervisory control of the arming scheme might be required to handle these issues and future work should focus on simulating realistic PMU data. Determining the optimal settings for the arming scheme will also likely require an extensive simulation study involving thousands of contingencies. The results of these simulations should be verified using WECC-approved base cases to eliminate any sources of error in the MiniWECC model.

5. The MicroWECC

5.1. Background

Real-time digital simulators (RTS) are simulation tools composed of hardware and software that allow for power system simulations to be conducted in real-time. Real-time refers to the simulator's ability to solve system equations for one time-step at the same speed as a real-world clock. This ensures the simulated power system model runs at the same rate as the physical system. This is in contrast to non-real time simulators, or offline simulators, where the amount of real time required to perform all necessary calculations in a time-step may be longer or shorter than the length of the time-step [8].

In addition to providing a reliable and accurate study of power systems, there are several other benefits to real-time simulation. One is that physical devices can be connected directly to the simulator to be tested. This is referred to as hardware-in-the-loop (HIL) simulation and is primarily used to prototype and test protective relays or controllers [9]. Another advantage of RTS simulations is that they are performed much quicker allowing for engineers to complete studies more efficiently. However, a significant drawback to RTS systems is that they often impose serious restrictions on the size of systems that can be modeled due to hardware and computational limitations.

5.2. Problem Statement

The client for this research project, Bonneville Power Association, was seeking to model the Western Interconnect in a RTDS simulator, which is the trademark name for the RTS simulator developed by RTDS Technologies. Their goal was to use the RTDS simulator to test new control systems for use on the PDCI, however, the MiniWECC was too complex to model in

the simulator. The hypothesis was that power system reduction techniques could be applied to the MiniWECC to design a reduced-order model of the Western Interconnect.

The objective of this project was to design a new model, termed the MicroWECC, with less than 10 machines because the client estimated this was the maximum system size the RTDS could handle. The MicroWECC would also need to retain the approximate impedance, generation, and dynamic properties of the MiniWECC.

5.3. Procedure

The overall procedure followed in this project is outlined in Figure 31. First the MicroWECC would be developed in PST using standard model reduction techniques. The accuracy of the model would then be determined by performing dynamic analysis. Next the model would be ported to the commercial software PSLF. This was done because the client was unfamiliar with PST models. Finally, approximate model parameters would be proposed for the RTDS simulator.

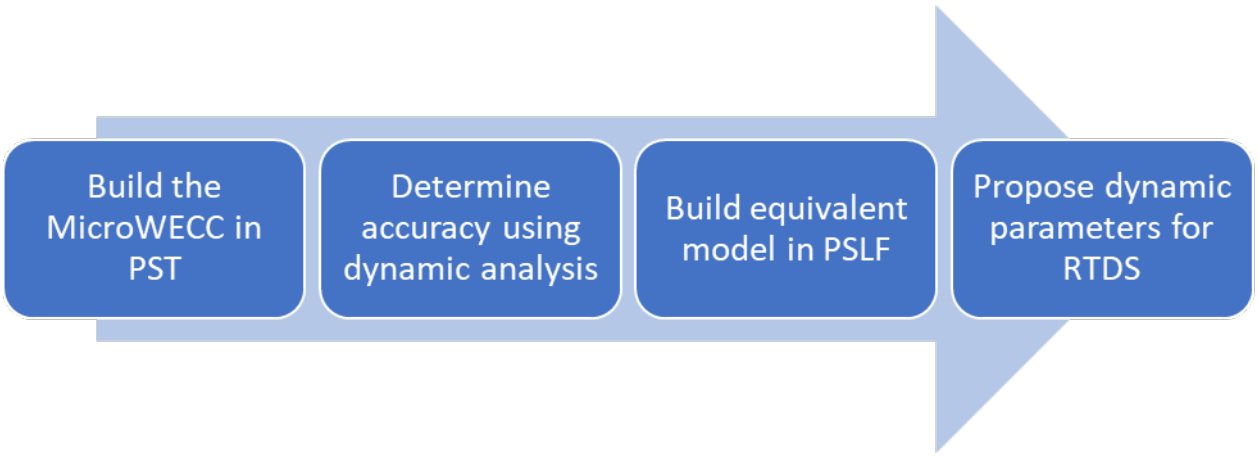

Figure 31. Procedure for Project 2

6. MicroWECC Design

The MicroWECC was developed using the general procedure shown in Figure 32. First generators within an area were grouped together by combining inertia values found in the MiniWECC. Next, transmission paths were built to connect between areas. These were calculated using reactance and resistance values from the MiniWECC model. Afterwards, generation and loads were adjusted to approximate MiniWECC nominal values and then simulated to ensure the system was transient stable. Finally, the process was repeated for the next area. Three major areas were used, so these steps were performed three times.

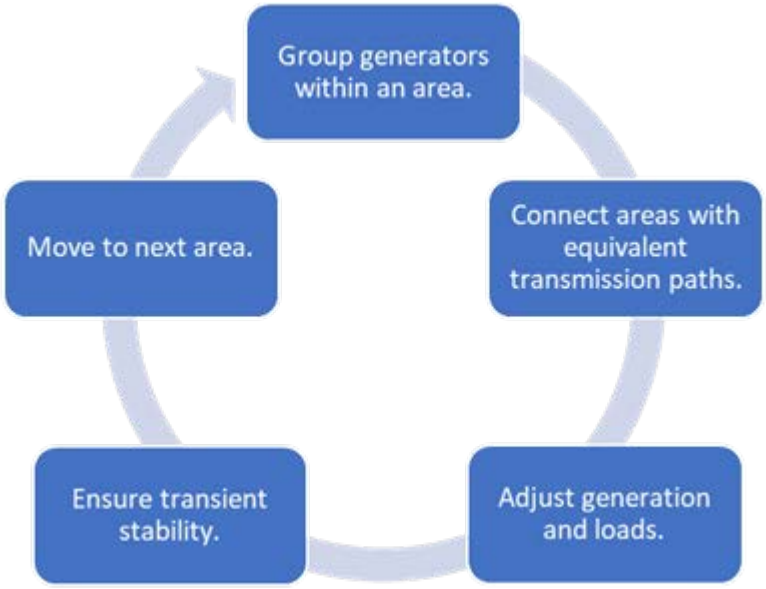

Figure 32. MicroWECC Design Procedure

The *overall* process of creating the MicroWECC was performed twice, resulting in two versions of the model. The first yielded positive results but the second used a more refined approach to produce a better model. The in-depth process to create the second version is explained in the following sections, however the only notable differences between the two

versions include an additional transmission line connection between Gen 1 and Bus 6 of the MiniWECC, slightly different generation, load, and transmission line values, and slightly different generator sizes. The exact differences can be found by comparing versions 1 and 2 found in Appendix N and Appendix O, respectively.

6.1. Colstrip to Alberta

The first transmission path that was calculated connects from the Colstrip generator (Gen 1) to a bus in northeast Washington and from that bus to a generator in Alberta (Gen 2). In the MiniWECC shown in Figure 33 this corresponds to the path from Gen 14 to Bus 6 and from Bus 6 to Gen 34. Intermediate busses 4-7 were added to represent substations at Garrison, Taft, Bell, and Selkirk, respectively. Sizes for MicroWECC generators 1 and 2 were copied from corresponding Gens 14 and 34 in the MiniWECC model as shown in Table II. A load was also added to MicroWECC Bus 8 to match the load on Bus 119 in the MiniWECC. The resulting system is shown in Figure 34. Exciter and governor parameters were taken from the corresponding MiniWECC generators and a power system stabilizer (PSS) was added at Gen 1 to match the MiniWECC. Transformer values for Gens 1 and 2 were taken directly from the MiniWECC model while transmission line resistance and reactance values were either taken directly or calculated using series and parallel rules as necessary. The resulting values for transformers and transmission lines are recorded in Table III.

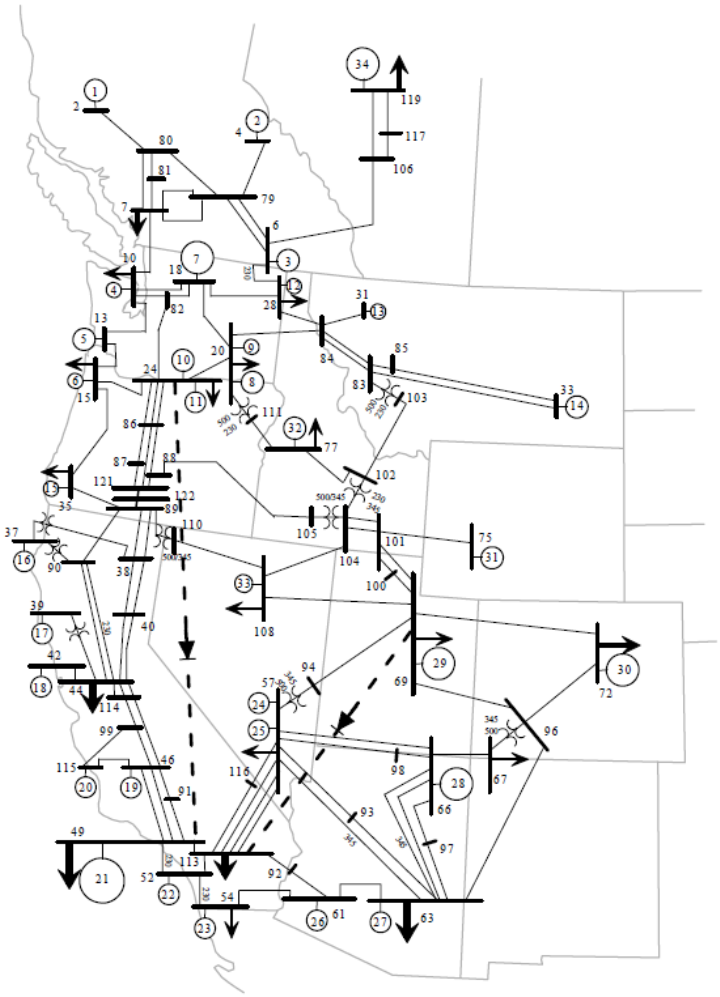

Figure 33. MiniWECC One-Line Diagram

Table II: Machine Size and Type for Gen 1 and Gen 2

Description	MicroWECC Gen #	MiniWECC Gen #	Base (MW)	Inertia (H)	Type
Colstrip	1	14	2300	6.5	Steam
Alberta	2	34	13000	6.5	Steam

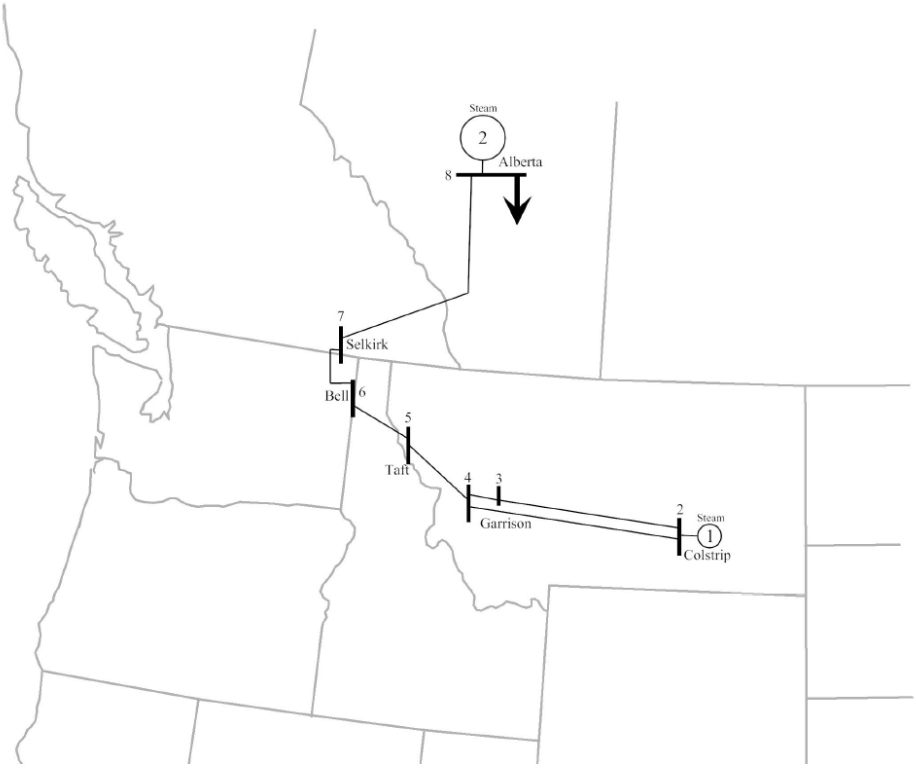

Figure 34. Gen 1 to Gen 2 MicroWECC One-Line Diagram

Table III: Transformer and Transmission Line Values from Gen 1 to Gen 2

Description	MicroWECC Line (Bus#-Bus#)	MiniWECC Line (Bus#-Bus#)	Resistance (PU)	Reactance (PU)
Colstrip XFMR	1-2	32-33	0.000000	0.00521740
Alberta XFMR	8-9	119-118	0.000000	0.00092308
Alberta load XFMR	8-10	119-120	0.000000	0.0010753
Colstrip-Garrison Line 1	2-3	33-85	0.002800	0.0350400
Colstrip-Garrison Line 2	2-4	33-83	0.003500	0.0438000
Garrison	3-4	85-83	0.000700	0.0087600
Garrison - Taft	4-5	(83-84)‖(83-84)	0.000600	0.0111000
Taft-Bell	5-6	84-28	0.001200	0.0202000
Bell-Selkirk	6-7	28-6	0.003125	0.0375000
Selkirk-Alberta	7-8	(6-106)+(106-119)‖ (106-117-119)	0.004600	0.0741000

The generation values for Gen 1 and Gen 2 as well as the load at Bus 8 were set to nominal values found in the MiniWECC model. Shunt capacitors were added to busses as

needed to fix under or over bus voltages. Gen 2 was designated to be the swing bus and a simulation disconnecting the line between bus 3 and 4 at t=1.0s, and then opening the line, was run to ensure the system was stable. Generator speeds from this simulation are shown in Figure 35.

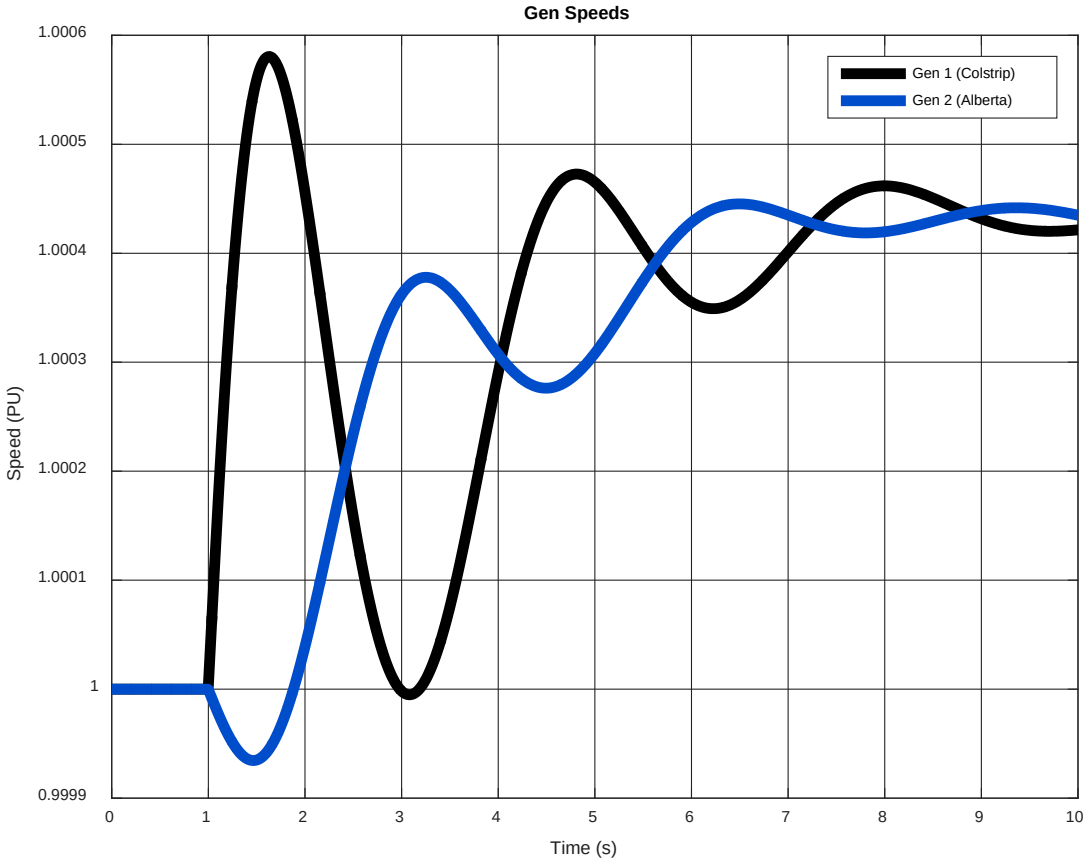

Figure 35. Gens 1 and 2 Test Simulation

6.2. Taft to British Columbia

Next the transmission paths connecting busses 5, 21, and 18 (Taft, Big Eddy, and British Colombia substations) were calculated and added to the system. Intermediate busses 16, 15, 11, and 12 were also added to represent substations at Seattle, Chief Joseph, Ashe, and Schultz, respectively. Gens 3, 4, 5, and 6 were added to represent hydro generators at Grand Coulee (GC),

British Colombia (BC), John Day, and Upper Columbia. Machine sizes were calculated by grouping MiniWECC generators in the same area together as specified in Table IV. Loads were added at busses 18, 16, and 21. The new system is shown in Figure 36. Exciter and governor models were copied from corresponding MiniWECC generators. A PSS was added at Gen 4 to match the MiniWECC model. Transformer values for generators and loads were taken from corresponding locations while transmission line resistance and reactance values were calculated using parallel and series rules. The resulting values are detailed in Table V.

Table IV: Machine Size and Type for Gens 3-6

Description	MicroWECC Gen #	MiniWECC Gen #	Base (MW)	Inertia (H)	Type
GC	3	7	8300	5.0	Hydro
BC	4	1+2+4	10800	5.0	Hydro
Upper Columbia	5	8+9	5200	5.0	Hydro
John Day	6	10+11	8900	5.0	Hydro

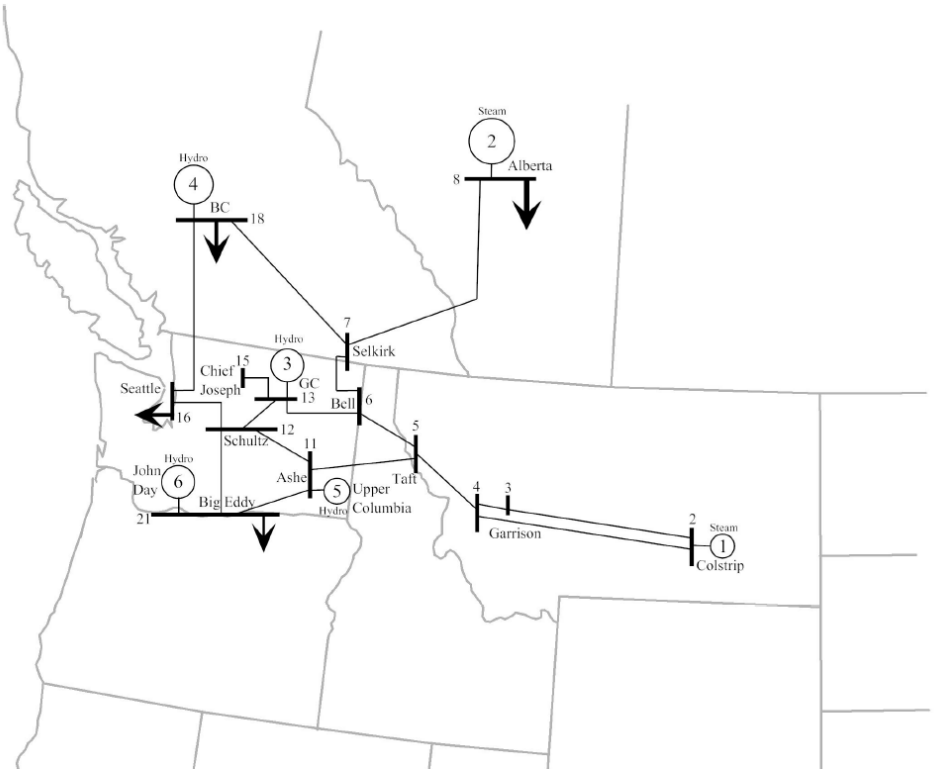

Figure 36. Gen 1 to Gen 6 and Gen 4 MicroWECC One-Line Diagram

Table V: Transformer and Transmission Line Values from Bus 5 to 18

Description	MicroWECC Line (Bus#-Bus#)	MiniWECC Line (Bus#-Bus#)	Resistance (PU)	Reactance (PU)
GC XFMR	13-14	17-18	0.000000	0.0014458
BC XFMR	18-19	1-2	0.000000	0.0026667
Up. Columb. XFMR	23-11	19-20	0.000000	0.0033333
John Day XFMR	24-21	23-24	0.000000	0.0024000
Seattle load XFMR	16-17	11-10	0.000000	0.0016667
BC load XFMR	18-20	8-7	0.000000	0.0012500
Big Eddy load XFMR	21-22	26-24	0.000000	0.0200000
Selkirk to BC	7-18	(6-79)‖(6-79)+ (79-80)	0.002761	0.028500
Bell-GC	6-13	28-18	0.000640	0.014560
Taft - Ashe	5-11	84-20	0.002150	0.031350
Ashe - Schultz	11-12	20-18	0.001100	0.020700
Ashe - Big Eddy	11-21	20-24	0.000000	0.002000
Schultz - GC	12-13	18-82	0.000400	0.006300
GC - Chief Joseph	13-15	18-82	0.000400	0.006300
Schultz - Seattle	12-16	(18-10)‖(18-10)	0.000650	0.012550
Seattle - BC	16-18	(10-7)+(7-80)‖ (7-81-80)	0.010789	0.036920
Schultz - Big Eddy	12-21	82-24	0.000343	0.007409

Gen 3 was designated the swing bus to match the MiniWECC and generation and load values at Bus 18 were adjusted to the nominal values found in the MiniWECC. Generation at Gen 5 and 6 were set to 50% of their nominal values. Loads at Bus 16 and 21 were adjusted high enough to absorb the excess generation. Shunt capacitors were adjusted and added to busses as needed to fix undervoltage problems. A simulation disconnecting the line between Bus 3 and 4 at t=1.0s, and then opening the line, was run to ensure the system was stable. Generator speeds from this simulation are shown in Figure 37.

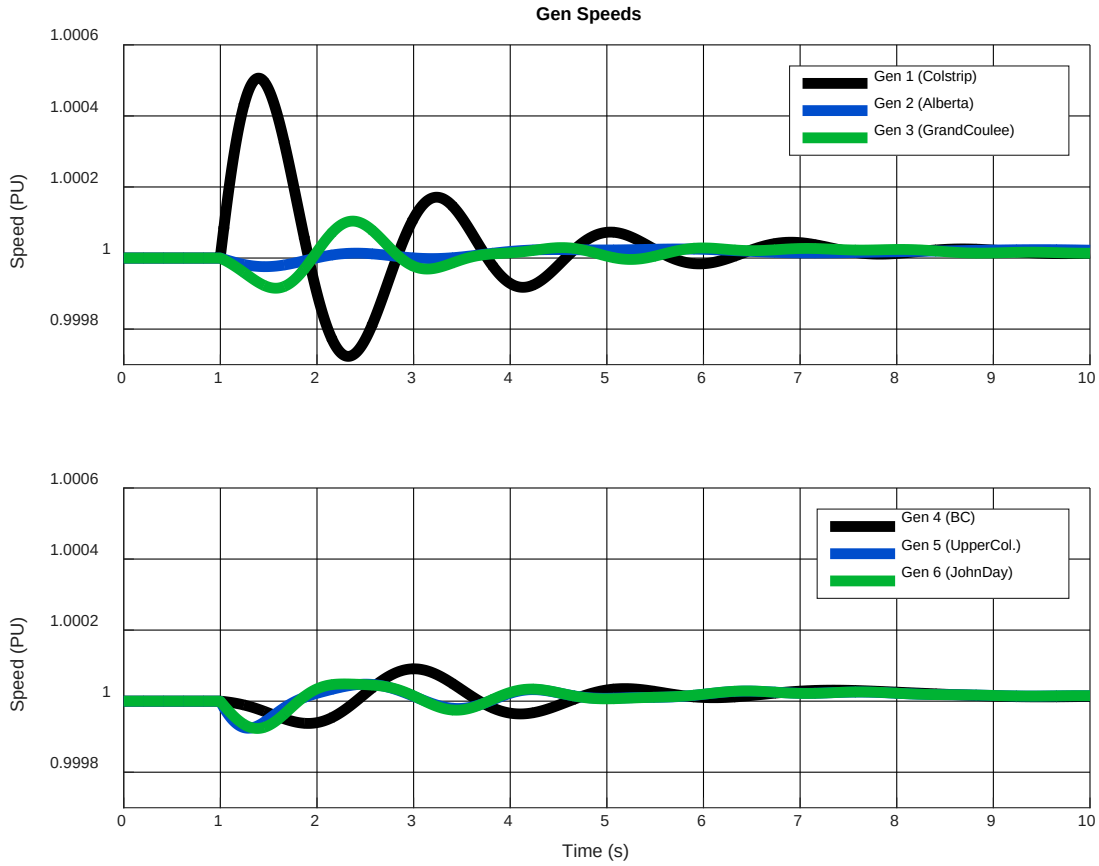

Figure 37. Gens 1-6 Test Simulation

6.3. Big Eddy to Vincent and Mead

Next the transmission path connecting the existing system with busses 30 and 33 (Vincent and Mead substations) was added. This path is comprised of two parallel AC lines and a high-voltage DC line, the Pacific DC Intertie (PDCI), that both run between Bus 21 and 30 as well as another transmission line connecting Bus 30 to 33. Additionally, a high-impedance transmission path was added to connect Bus 33 back to Bus 4. Intermediate busses 27, 25, and 26 were added to represent substations at San Francisco, Captain Jack, and Malin, respectively. Gens 7, 8, and 9 were added to represent power plants at Moss Landing, Palo Verde, and Navajo. Generator sizes were calculated according to Table VI. Loads were also added at busses 30 and

33. These additions completed the MicroWECC model and the resulting system is shown in Figure 38. Exciter and governor values were taken from the corresponding MiniWECC generators. Transformer and transmission lines values were calculated using parallel and series rules and the results are detailed in Table VII.

Table VI: Machine Size and Type for Gens 7-9

Description	MicroWECC Gen #	MiniWECC Gen #	Base (MW)	Inertia (H)	Type
Moss Landing	7	16+17+18	10000	6.5	Gas
Palo Verde	8	19+20+21+22+23	29900	6.5	Steam
Navajo	9	24+25+28+29	28160	6.5	Steam

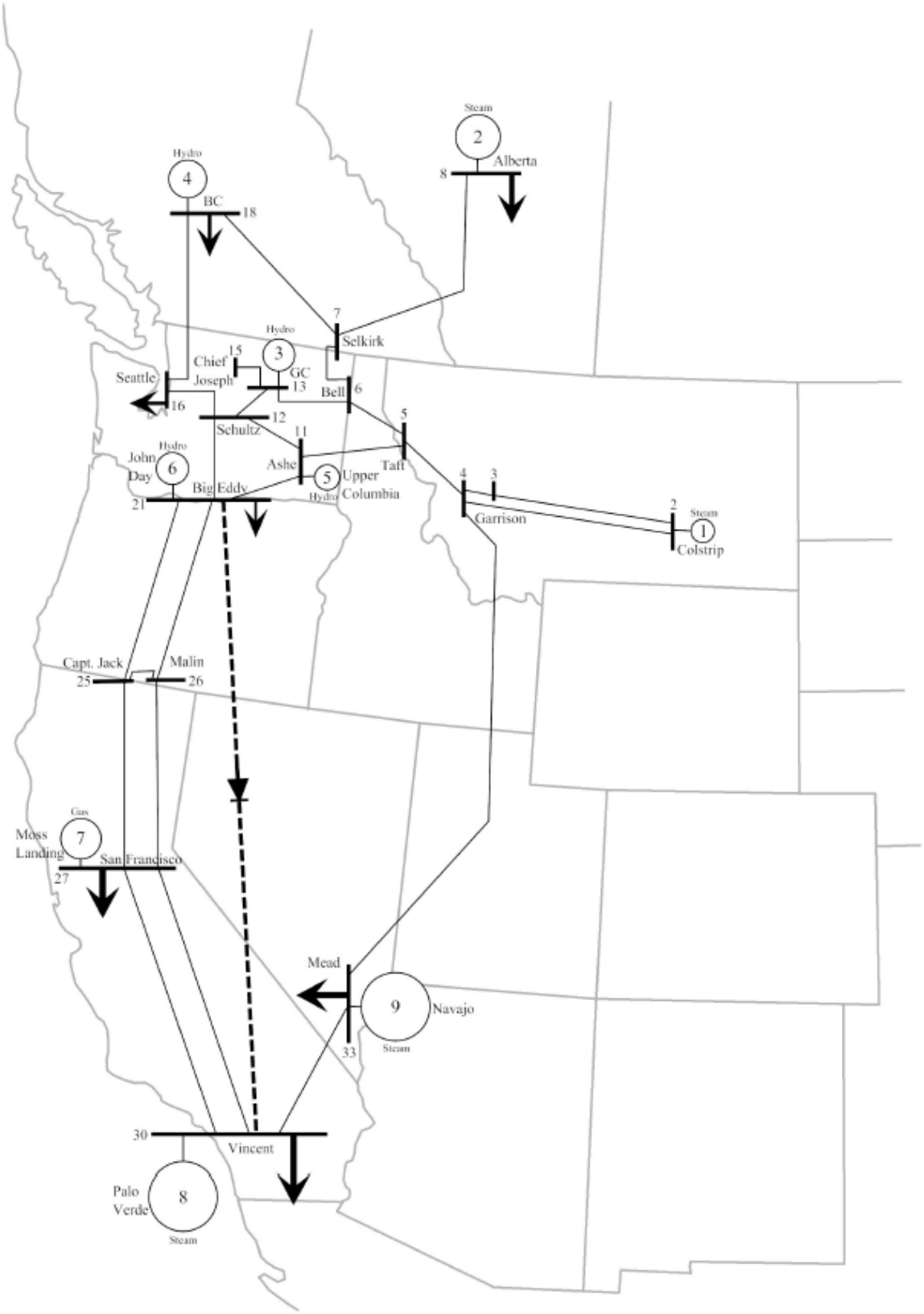

Figure 38. MicroWECC One-Line Diagram

Table VII: Transformer and Transmission Line Values from Bus 21 to Bus 33 and Bus 4

Description	MicroWECC Line (Bus#-Bus#)	MiniWECC Line (Bus#-Bus#)	Resistance (PU)	Reactance (PU)
Moss Landing XFMR	28-27	41-42	0.000000	0.00115380
Palo Verde XFMR	31-30	48-49	0.000000	0.00067416
Navajo XFMR	34-33	58-57	0.000000	0.00375000
San Fran. load XFMR	29-27	43-44	0.000000	0.00043103
Vincent load XFMR	32-30	50-49	0.000000	0.00076923
Mead load XFMR	35-30	56-57	0.000000	0.00125000
Big Eddy-Captain Jack	21-25	2*([(24-86)||(24-86)|| (24-86)]+(86-87-121)|| (86-121)||(86-88-121)]|| [24-15-35-89])	0.0015442	0.0142856
Big Eddy-Malin	21-25	2*([(24-86)||(24-86)|| (24-86)]+(86-87-121)|| (86-121)||(86-88-121)]|| [24-15-35-89])	0.0015442	0.0142856
Captain Jack-Malin	25-26	NA	0.0000000	0.0001000
Captain Jack-San Francisco Line 1	25-27	2*([(89-38)||(89-38)+ (38-40)||(38-40)+ (40-44)||(40-44)]|| [(89-90)+(90-44)|| (90-44)])	0.0019474	0.0153720
Malin-San Francisco Line 2	26-27	2*([(89-38)||(89-38)+ (38-40)||(38-40)+ (40-44)||(40-44)]|| [(89-90)+(90-44)|| (90-44)])	0.0019474	0.0153720
San Francisco-Vincent Line 1	27-30	2*[(44-114)||(44-114)|| (44-114)+(114-46)|| [(114-99)+(99-46)|| (99-115-46)]+(46-49)|| (46-49)||(46-91-49)]	0.0026278	0.0317680
San Francisco-Vincent Line 2	27-30	2*[(44-114)||(44-114)|| (44-114)+(114-46)|| [(114-99)+(99-46)|| (99-115-46)]+(46-49)|| (46-49)||(46-91-49)]	0.0026278	0.0317680
Vincent-Mead	30-33	(113-57)||(113-57)|| (113-57)||(113-116-57)	0.000322	0.00433
Mead-Garrison	33-4	(83-103-102-104)+ (104-101)||(104-101)+ (101-69)||(101-100-69)+ (69-94-57)	0.0473	0.2062

The PDCI was modeled as a 2850 MW load at the Bus 21 and a 2850 MW power injection at Bus 30 which were values taken from the MiniWECC model. Generation values were set to the sum of generation from the respective generators they represent in the

MiniWECC. The exception to this was Gen 9 where a value of 10GW was experimentally chosen due to larger generation values negatively impacted the stability and modal shapes of the system. The loads in the MicroWECC were then adjusted to give similar power flows as the MiniWECC. Specifically, it was ensured that roughly 2800 MW of power was transmitted from Big Eddy to Vincent along the AC intertie. Shunt capacitors were adjusted or added to busses as needed to fix voltage problems. A simulation disconnecting the line between bus 3 and 4 at t=1.0s, and then opening the line, was run to ensure the system was stable. The same simulation was run by disconnecting the line between bus 85 and 83 and then opening the line in the MiniWECC. Generators speeds were then compared against the corresponding generators. Speeds for the MicroWECC generators 1-3 and their corresponding MiniWECC generators are shown in Figure 39. Speeds for MicroWECC generators 4-6 and their corresponding MiniWECC generators are shown in Figure 40. Speeds for MicroWECC generators 7-9 and their corresponding MiniWECC generators are shown in Figure 41. Solid lines represent the MicroWECC generator speeds and dotted lines represent the corresponding MiniWECC generator speeds.

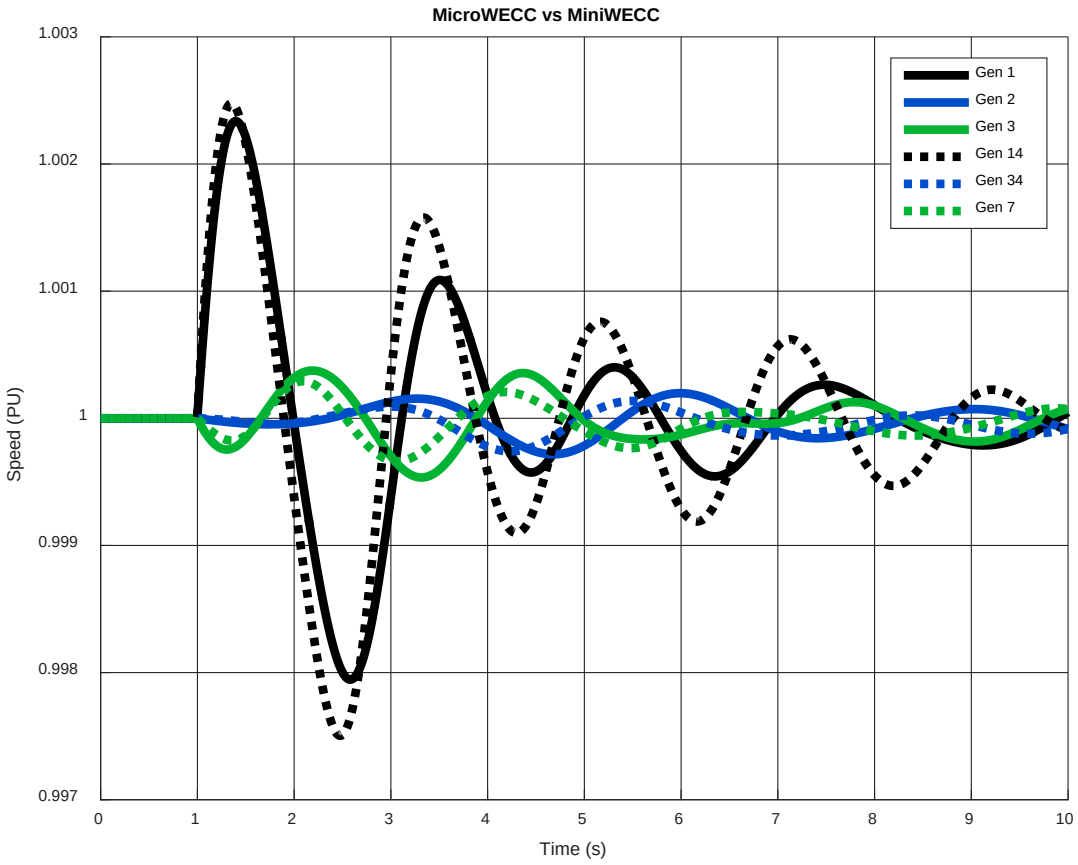

Figure 39. MicroWECC Gens 1, 2, and 3 vs MiniWECC Gens

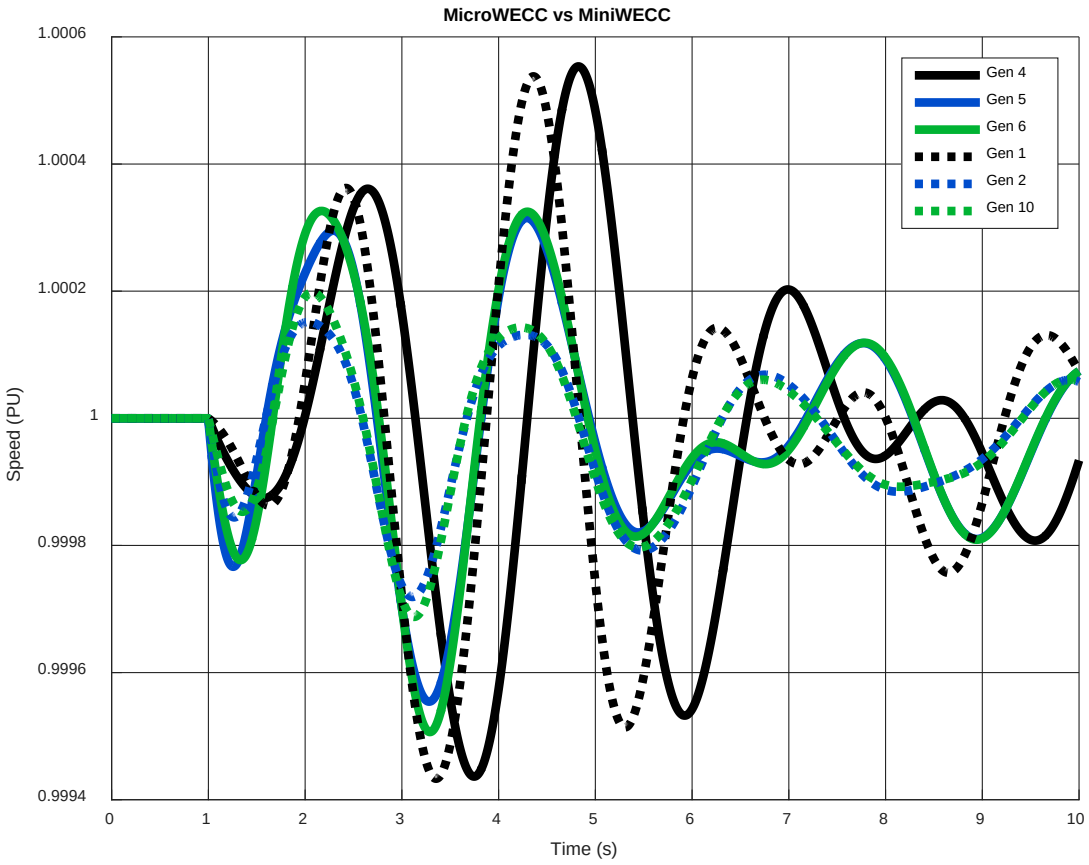

Figure 40. MicroWECC Gens 4, 5, and 6 vs MiniWECC Gens

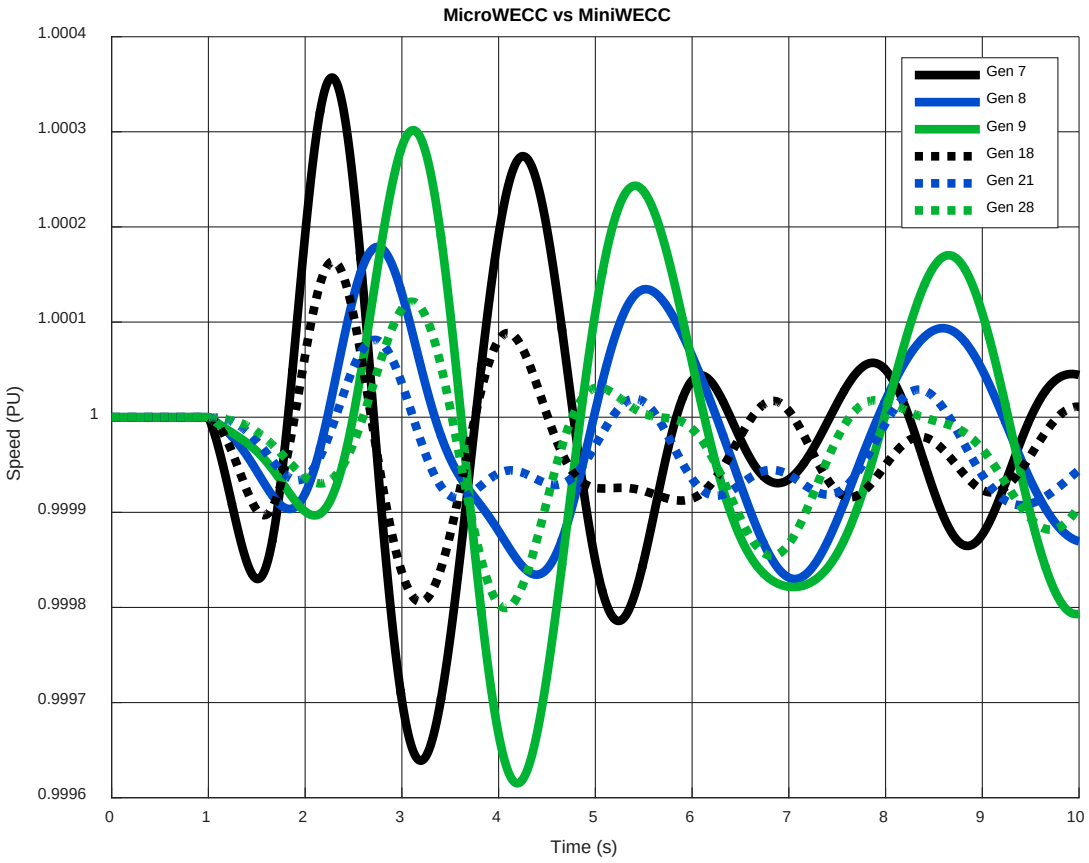

Figure 41. MicroWECC Gens 7, 8, and 9 vs MiniWECC Gens

From the results of this simulation it is clear the two models do not match exactly in terms of frequency and amplitude of their generator speed responses. This is to be expected however, because the generators in the MicroWECC were formed as composites of multiple generators in the MiniWECC, often from generators that were separated by transmission lines. This resulted in the MicroWECC generators being much larger, especially Gens 7, 8 and 9, which also explains why their speed oscillations are much larger than the speed oscillations in the MiniWECC. Also, although care was taken in making transmission paths equivalent, to reduce the model order, many transmission paths, generators and loads were not included the MicroWECC. The combination of these differences account for the different speed responses. To

verify the MicroWECC model against the MiniWECC, a more accurate method of dynamic analysis would need to be performed: modal analysis.

7. Modal Analysis

Power systems, like all dynamic systems, exhibit natural oscillatory behavior when excited. These oscillations occur at resonant frequencies and in specific patterns of motion referred to as modes. The number of unique modes in a power system is equal to the number of generators. Modal analysis is the process of estimating the frequencies and shapes of modes in a given system. The method of modal analysis that is described and performed in this section is known more specifically as eigenanalysis.

Studying the oscillatory modes of a power system is important for several reasons. First, power system oscillations, if not correctly damped, can directly result in system collapse. Second, comprehensive understanding of the oscillatory modes in a system is required to design control methods that increase, and not decrease, damping. The reason eigenanalysis was performed on the MicroWECC was to compare the modes of the MicroWECC model to the modes of the MiniWECC model in order to determine the accuracy of the model-order reduction.

The first step in performing eigenanalysis is to linearize the first-order differential equations of the system. Once that has been done, the system can be represented in state-space form. The general form is shown in (6):

$$\begin{aligned} \dot{\underline{x}}(t) &= A\underline{x}(t) + B\underline{q}(t) \\ \underline{y}(t) &= C\underline{x}(t) + D\underline{q}(t) \end{aligned} \tag{6}$$

Where,

$\underline{x}$ is the state vector with length n (number of states)

$\dot{\underline{x}}$ is the derivative of $\underline{x}$ with respect to time

A is the state matrix with dimensions n by n

B is the input matrix with dimensions n by n_i (number of inputs)

$\underline{q}$ is the input vector with length n_i

$\underline{y}$ is the output vector with length n_o (number of outputs)

C is the output matrix with dimensions n_o by n

D is the feed forward matrix with dimensions n_o by n_i

7.1. Eigenvalues and Eigenvectors

Mathematically, the modes of a system are described by the eigenvalues and eigenvectors of the matrix *A*. The eigenvalues of the state matrix can be determined by finding the roots of the characteristic polynomial and are defined by (7):

$$|A - \lambda_i I| = 0 \tag{7}$$

Where *I* is the identity matrix with dimensions n by n and λ_i is the *i*th eigenvalue with *i* ranging from 1 to n. For each eigenvalue there is a corresponding right and left eigenvector that satisfy (8) and (9):

$$A\underline{u}_i = \lambda_i \underline{u}_i \tag{8}$$

$$\underline{v}_i A = \lambda_i \underline{v}_i \tag{9}$$

Where $\underline{u}_i$ is the *i*th right eigenvector with dimensions n by 1 and v_i is the *i*th left eigenvector with dimensions 1 by n. The eigenvectors can be combined into n by n square matrices in (10) and (11):

$$U = [\underline{u}_1 \quad \cdots \quad \underline{u}_n] \tag{10}$$

$$V = \begin{bmatrix} \underline{v}_1 \\ \vdots \\ \underline{v}_n \end{bmatrix} \tag{11}$$

U and *V* are inverses of each other as shown in (12) and (13), and have the orthonormal property $VU = I$ [10].

$$V = U^{-1} \tag{12}$$

$$U = V^{-1} \tag{13}$$

Now let *z* be a new variable defined by (14):

$$\underline{z}(t) = V\underline{x}(t) \tag{14}$$

Solving for $\underline{x}$ using (12) and (13) results in (15):

$$\underline{x}(t) = U\underline{z}(t) \tag{15}$$

This can be rewritten as (16):

$$\underline{x}(t) = \sum_{i=1}^{n} z_i(t)\underline{u_i} \tag{16}$$

Solving this equation for an arbitrary state $x_k(t)$ results in (17):

$$x_k(t) = z_1(t)u_{1,k} + z_2(t)u_{2,k} \dots z_n(t)u_{n,k} \tag{17}$$

To solve for the outputs of the system, plug (15) into (6) to get (18):

$$U\dot{\underline{z}}(t) = AU\underline{z}(t) + B\underline{q}(t) \tag{18}$$

Multiply both sides by *V* to get (19):

$$\dot{\underline{z}}(t) = VAU\underline{z}(t) + VB\underline{q}(t) \tag{19}$$

Noting that $VAU = diag(\lambda_i)$, this reduces to (20) [10]

$$\dot{z}_i(t) = \lambda_i z_i(t) + \underline{v_i}B\underline{q}(t) \tag{20}$$

Transforming into the Laplace domain results in (21):

$$sZ_i(s) = \lambda_i Z_i(s) + \underline{V_i} B \underline{Q}(s) + Z_i(0) \tag{21}$$

Solving for $Z_i(s)$ results in (22):

$$Z_i(s) = \frac{Z_i(0)}{s - \lambda_i} + \frac{\underline{V_i} B}{s - \lambda_i} \underline{Q}(s) \tag{22}$$

Assuming the eigenvalue is complex with the form of (23):

$$\lambda_i = \alpha_i + j\omega_i \tag{23}$$

The time domain solution will take the form in (24) for any input:

$$z_i(t) = Ke^{\alpha_i t}\cos(\omega_i t + \varphi) \tag{24}$$

Where the values of K and φ depend on the input and the observed output. In this form it is possible to see that eigenvalue determines both the damping and frequency of an oscillation. The oscillation will grow, remain constant, or decay depending on the value of the real part of the complex eigenvalue, while the frequency in rad/sec is specified by the imaginary part [11].

Equation (17) shows that the right eigenvectors act as weights for the states of the system. Specifically, the magnitude of $u_{i,k}$ determines the amplitude of mode i to show up in state x_k. The angle of $u_{i,k}$ determines the phasing of ith mode in state x_k. The phase determines whether a given state is oscillating with or against a different state. Because of the information the right eigenvectors provide, they are often referred to as the "mode shape" [11]. It is common practice to normalize the eigenvector so that the largest value has a magnitude of 1 and an angle of 0.

It should be noted that some systems, such as power systems, have a real eigenvalue (no imaginary part) that describes a non-oscillatory mode. This is sometimes referred to as the common-mode. The common-mode can be considered the steady-state displacement the system

undergoes due to a disturbance and is not relevant to the discussion of oscillatory modes. It is also worth mentioning that complex eigenvalues always come in conjugate pairs. For the purposes of this research, only the eigenvalues with positive imaginary components were considered.

7.2. Eigenanalysis Example

Eigenanalysis can only be performed on a linear system. Power systems are inherently nonlinear systems but can be linearized around steady-state operating conditions. When linearized, a power system can be considered as a system of masses attached by springs. Generators are analogous to masses and transmission lines are analogous to springs.

To illustrate how eigenanalysis is performed; consider the mass-spring-damper system shown in Figure 42. This system consists of three masses connected by two springs, k1 and k2, and two dampers, d1 and d2. Additionally, there is friction between each mass and the ground, d0. An input force, f(t), is applied to the first mass.

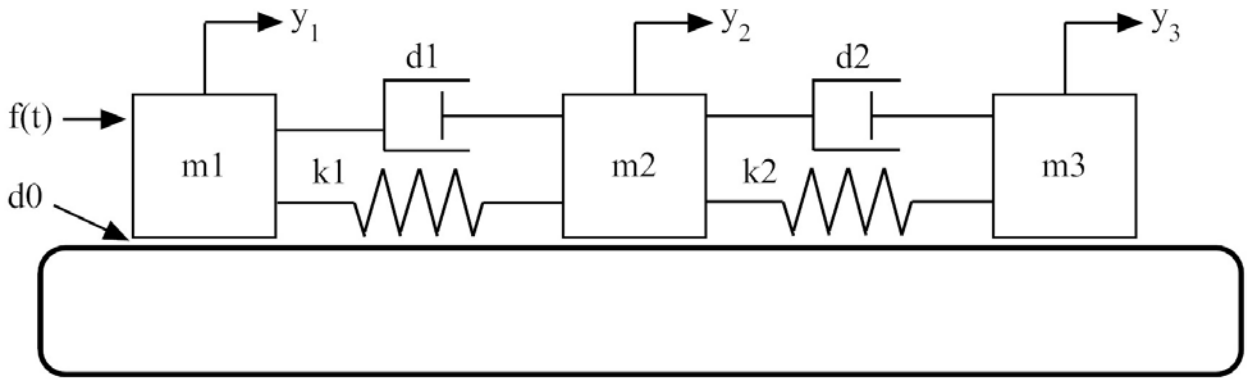

Figure 42. 3 Mass, 2 Spring-Damper System

To apply eigenanalysis, this system must be built in state-space form first. The desired states of this system are the displacement and velocity of each mass. The displacement of masses 1, 2 and 3 will be designated y_1, y_2, and y_3 respectively, as shown in the figure. The velocities of

each mass are the derivatives of the displacement and therefore designated $\dot{y}_1$, $\dot{y}_2$, and $\dot{y}_3$. The resulting state vector is shown in (25):

$$\underline{x} = \begin{bmatrix} y_1 \\ \dot{y}_1 \\ y_2 \\ \dot{y}_2 \\ y_3 \\ \dot{y}_3 \end{bmatrix} \tag{25}$$

The derivative of the state vector, or $\underline{\dot{x}}$, is found by formulating the state equations of the system. The derivative state $\dot{x}_1$ is the derivative of displacement y_1, making it equal to x_2. Likewise, the derivative state $\dot{x}_3$ is equal to x_4, and the derivative state $\dot{x}_5$ is equal to x_6. The remaining derivative states $\dot{x}_2$, $\dot{x}_4$, and $\dot{x}_6$ represent the acceleration of each mass, or $\ddot{y}_i$ where i ranges from 1 to 3, and can be solved for using Newton's second law of motion which is defined in (26):

$$\sum F_i = m_i \ddot{y}_i \tag{26}$$

Where F_i represents all forces (input force and forces of springs and dampers) applied to the ith mass, m_i represents the ith mass value, and $\ddot{y}_i$ represents the ith mass's acceleration. Solving for the sum of forces applied to each mass and dividing by the ith mass, the full state equation becomes (27):

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \\ \dot{x}_5 \\ \dot{x}_6 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ -\frac{k1}{m1} & -\frac{(d0+d1)}{m1} & \frac{k1}{m1} & \frac{d1}{m1} & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ \frac{k1}{m2} & \frac{d1}{m2} & -\frac{(k1+k2)}{m2} & -\frac{(d0+d1+d2)}{m2} & \frac{k2}{m2} & \frac{d2}{m2} \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & \frac{k2}{m3} & \frac{d2}{m3} & -\frac{k2}{m3} & -\frac{(d0+d2)}{m3} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{m1} \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} f \tag{27}$$

Where

f is the input force applied to m_1

k1 and *k2* represent the spring constants of their respective spring

d1 and *d2* represent the viscous friction coefficients of their respective damper

d0 represents the viscous friction coefficient of the ground

The output of the system is determined by the state equation shown in (28):

$$\underline{y} = C\underline{x} + Df \tag{28}$$

Where *C* is set to the identity matrix I_6 to observe all states of $\underline{x}$ in the output $\underline{y}$, and *D* is set to a 6 by 1 matrix of zeros because there is no direct feedthrough in this system.

Now that the state equations of the system have been determined, the modes can be calculated, and the system can be simulated in MATLAB. The modes are calculated by finding the eigenvalues and right eigenvector of the state matrix which was determined by defining *A* and using it as the input to the MATLAB function *eig*. This function returns the right eigenvectors in a matrix *U* and the eigenvalues in the diagonal of matrix *D*. The eigenvalues can be extracted from *D* to the vector *l* using the function *diag*. Recall the mode frequency in rad/sec is found from the imaginary part of the eigenvalue. The mode frequency in Hz was therefore determined using (29):

$$f = \frac{\omega}{2\pi} = \frac{imag(l)}{2\pi} \tag{29}$$

The damping ratio of an eigenvalue is calculated by taking the ratio of the real part to the absolute value. The damping percent was therefore determined using (30):

$$\%Damping = \frac{real(l)}{abs(l)} * 100\% \tag{30}$$

The system was also simulated to show how the modes manifest as oscillations in the displacement and velocity states of the system. To accomplish this, *A*, *B*, *C*, *D* matrices were

defined and used as inputs to the MATLAB function *ss* to create a system object *G* that represented the continuous-time state-space model. Next a 30-second time vector, *t*, was created with a time-step of 0.001s. An input $f(t)$ was initialized to a vector of zeros with the length of the time vector and then the first value was set to the inverse of the time-step, which is 1000. This models an impulse that will instantaneously accelerate the first mass to a velocity of 1 during the first time step. The time response of the state-space system described by $f(t)$ and t can then be simulated using MATLAB's *lsim* function and plotted. In the following cases, the coefficients for the mass and springs were experimentally chosen so the results of the simulation had distinct modes of oscillation. In the first and second case, damping constants were held at 0 to illustrate the modes of oscillation without damping. The code for these cases can be found in Appendix P, Appendix Q, and Appendix R.

7.2.1. Case 1

The number of modes in this example is equal to the number of masses, 3. One mode however is the common mode. Thus, there are 2 oscillatory modes for this system. The coefficients for mass and springs were experimentally chosen so that both modes had distinct frequencies. The coefficients used in this simulation are shown in Table VIII.

Table VIII: Mass and Spring Constants for Case 1

Variable	Value
m1	1.00
m2	0.50
m3	20.0
k1	1.00
k2	10.0
d0	0.00
d1	0.00
d2	0.00

The mode frequencies and damping are shown in Table IX.

Table IX: Oscillatory Modes Frequency and Damping for Case 1

Mode (#)	Frequency (Hz)	%Damping
1	0.1554	0.00
2	0.7557	0.00

The first mode has a frequency of 0.155 Hz, so the period of oscillation is expected to be about 6.4 seconds. The second mode has a frequency of 0.756 Hz, so the period of oscillation is expected to be about 1.3 seconds. The first mode shape is shown in Figure 43 and the second mode shape is shown in Figure 44. Note that the amplitudes of some masses are difficult to see because they are relatively small compared to the dominant mass.

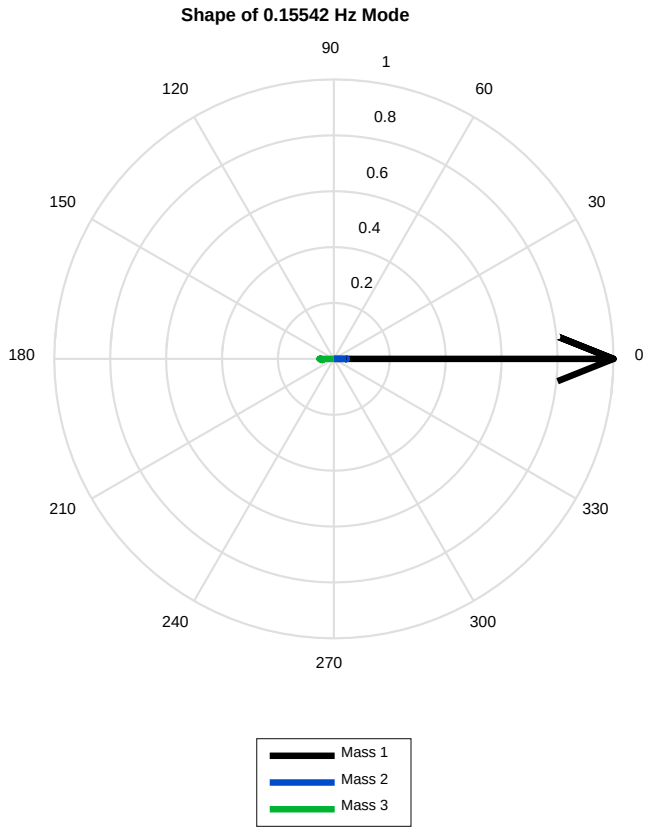

Figure 43. Case 1 Mode Shape for 0.155 Hz Mode

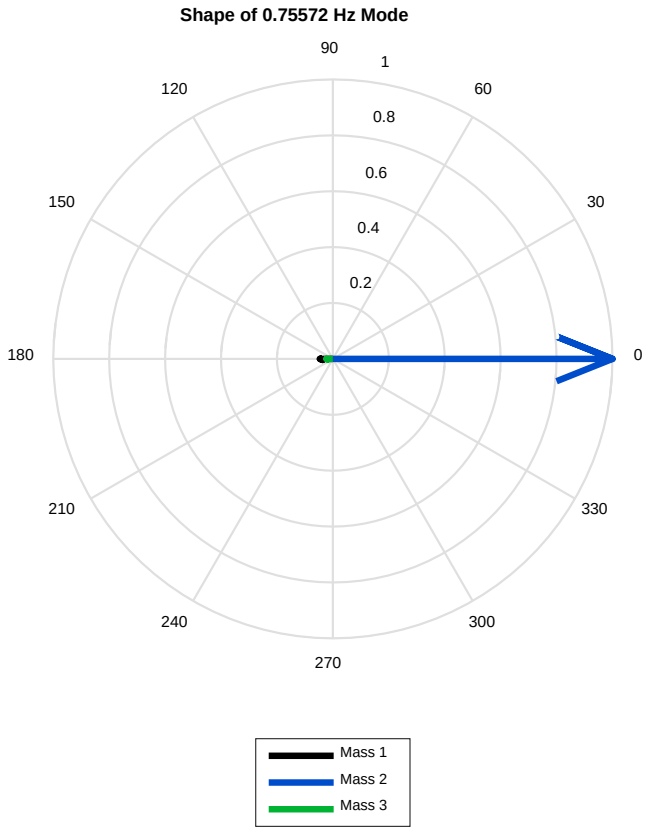

Figure 44. Case 1 Mode Shape for 0.756 Hz Mode

The 0.15 Hz mode shape can be interpreted as the first mass oscillating with the largest amplitude in phase with the second mass, which will have about 4% the peak-peak amplitude. The third mass oscillates 180° out of phase with the first and second mass with 5% the peak-peak amplitude of the first mass. The 0.75 Hz mode shape can be interpreted as the second mass oscillating with the largest amplitude 180° out of phase with the first and third mass which have about 5% and 2%, respectively, the peak-peak amplitude of the second mass. The simulation results for the mass displacements are shown in Figure 45. The mass velocities are shown in Figure 46.

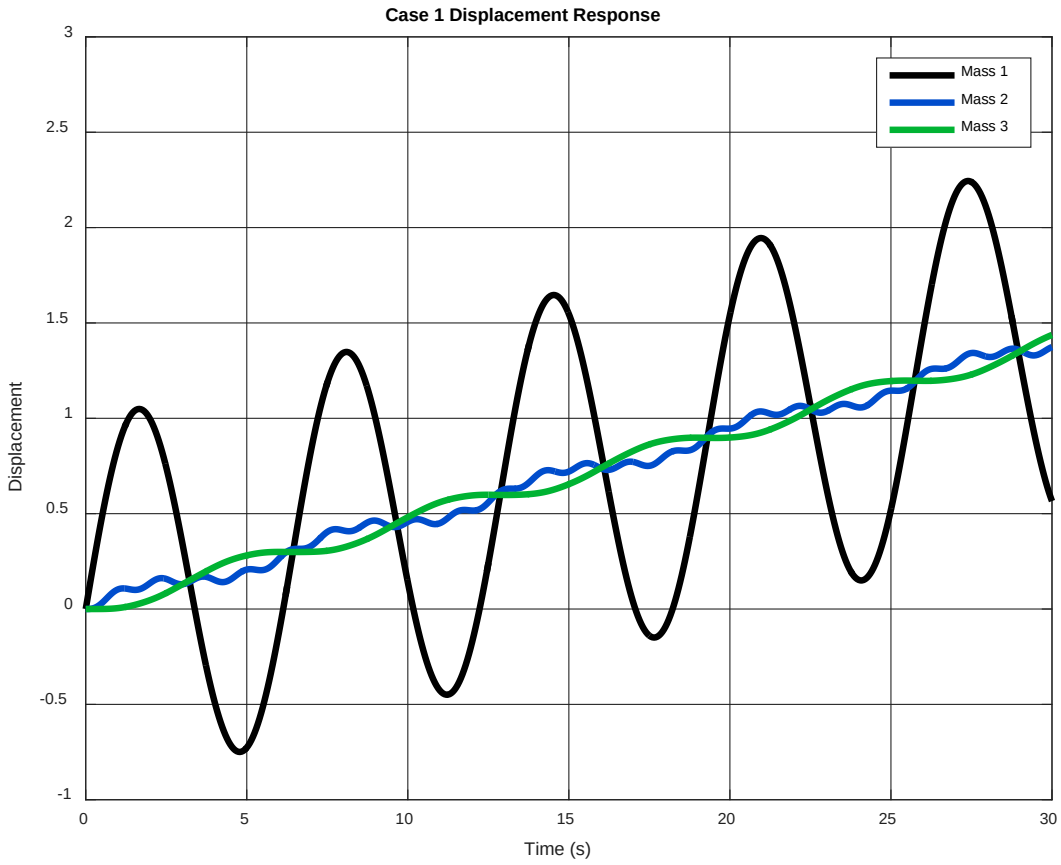

Figure 45. Case 1 Mass Displacement

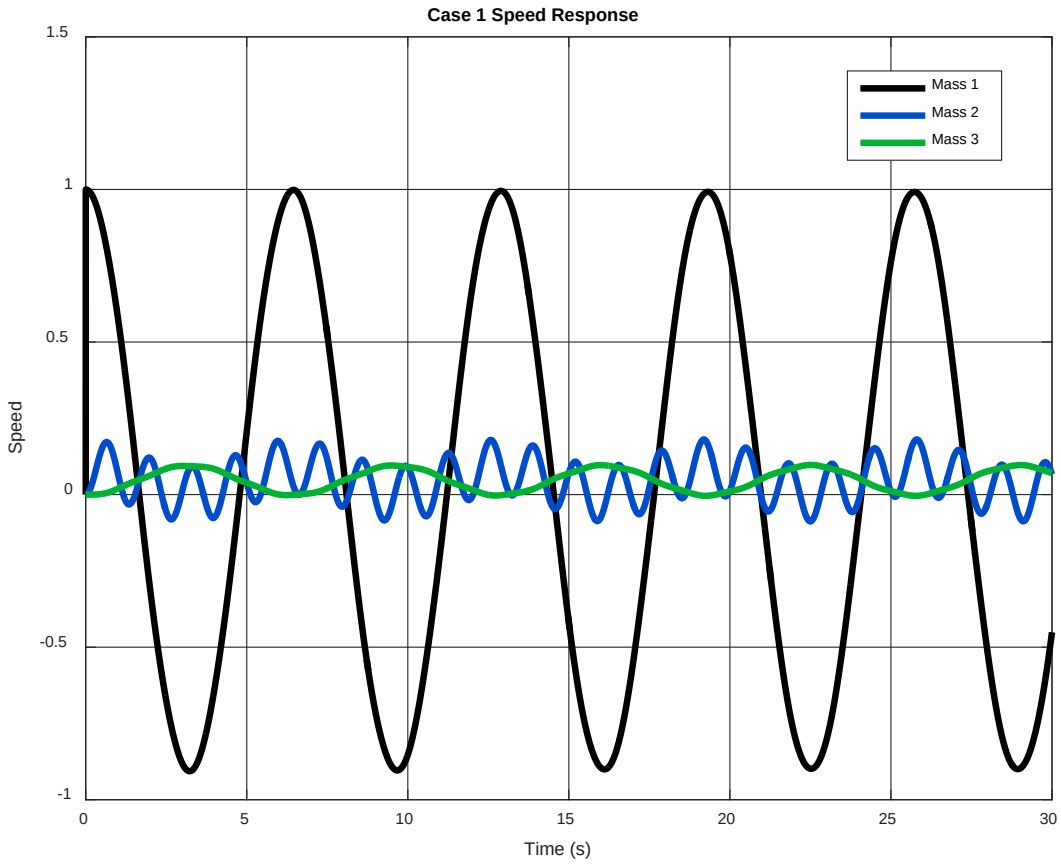

Figure 46. Case 1 Mass Speeds

The results of the simulation show that the first and third mass oscillate against each other with a period of roughly 6.4 seconds. This is clearly the 0.15 Hz mode. The shape of this mode predicted that third mass would oscillate against the first mass at 5% of the peak-peak amplitude which is almost exactly the difference in peak-peak amplitudes. The 0.15 Hz mode shape also predicted that the second mass would oscillate with the first mass at 4% of the peak-peak amplitude which is possible to see in the way the second mass oscillates up and down in phase with the first mass at this frequency. The simulation also shows the second mass is oscillating at a faster frequency that has a period of roughly 1.3 seconds. This is clearly the 0.75 Hz mode which had a mode shape that shared about 5% of the amplitude with the first and third

mass. Because the second mass oscillates with a small amplitude already, the amplitude effect of this mode is almost non-existent in the other mass even though it is present.

7.2.2. Case 2

Like the first case, the second case has no damping. Again, the mass and spring constants were experimentally chosen so that there were two distinct oscillatory modes, but less distinct than the first case. The mass and spring constants are shown in Table X

Table X: Mass and Spring Constants for Case 2

Variable	Value
m1	1.00
m2	5.00
m3	2.00
k1	1.00
k2	0.20
d0	0.00
d1	0.00
d2	0.00

The two oscillatory mode frequencies and damping for this system are shown in Table XI.

Table XI: Oscillatory Modes Frequency and Damping for Case 2

Mode (#)	Frequency (Hz)	%Damping
1	0.0579	0.000
2	0.1749	0.000

The first mode has a frequency of 0.058 Hz, so the period of oscillation is expected to be about 17.2 seconds. The second mode has a frequency of 0.175 Hz, so the period of oscillation is expected to be about 5.7 seconds. The shape of the first mode is shown in Figure 47 and for the second mode is shown in Figure 48.

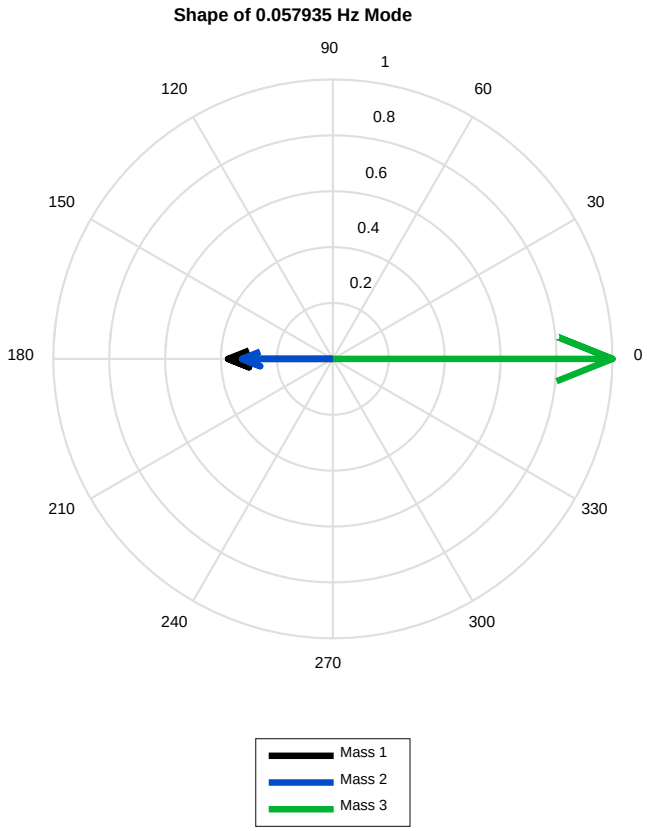

Figure 47. Case 2 Mode Shape for 0.0579 Hz Mode

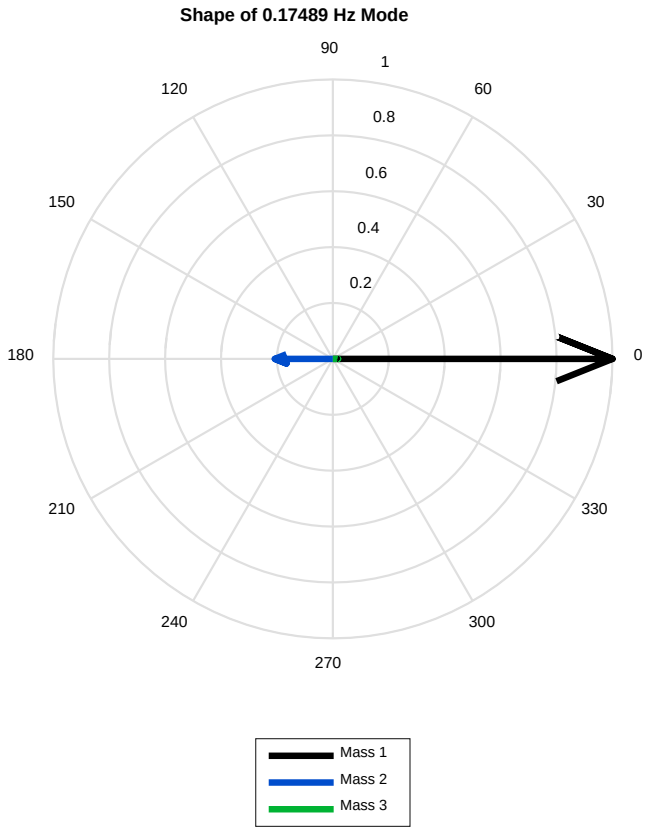

Figure 48. Case 2 Mode Shape for 0.1749 Hz Mode

The 0.058 Hz mode shape can be interpreted as the third mass oscillating with the largest amplitude 180° out of phase with the first and second mass, which will have about 37% and 33%, respectively, the peak-peak amplitude of the third mass. The 0.175 Hz mode shape can be interpreted as the first mass oscillating with the largest amplitude in phase with the third mass which has 2% the peak-peak amplitude. The second mass oscillates 180° out of phase with first and third mass at about 20% the peak-peak amplitude of the first mass. The simulation results for the mass displacements are shown in Figure 49. The mass velocities are shown in Figure 50.

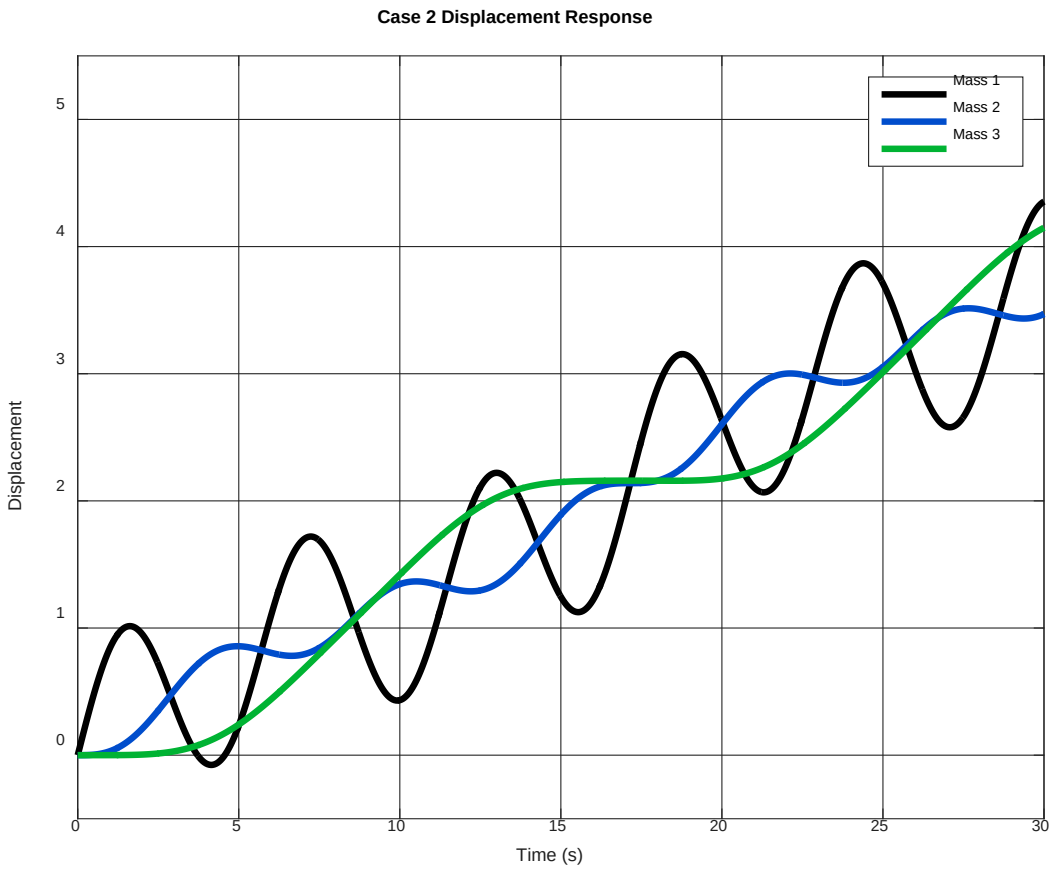

Figure 49. Case 2 Mass Displacement

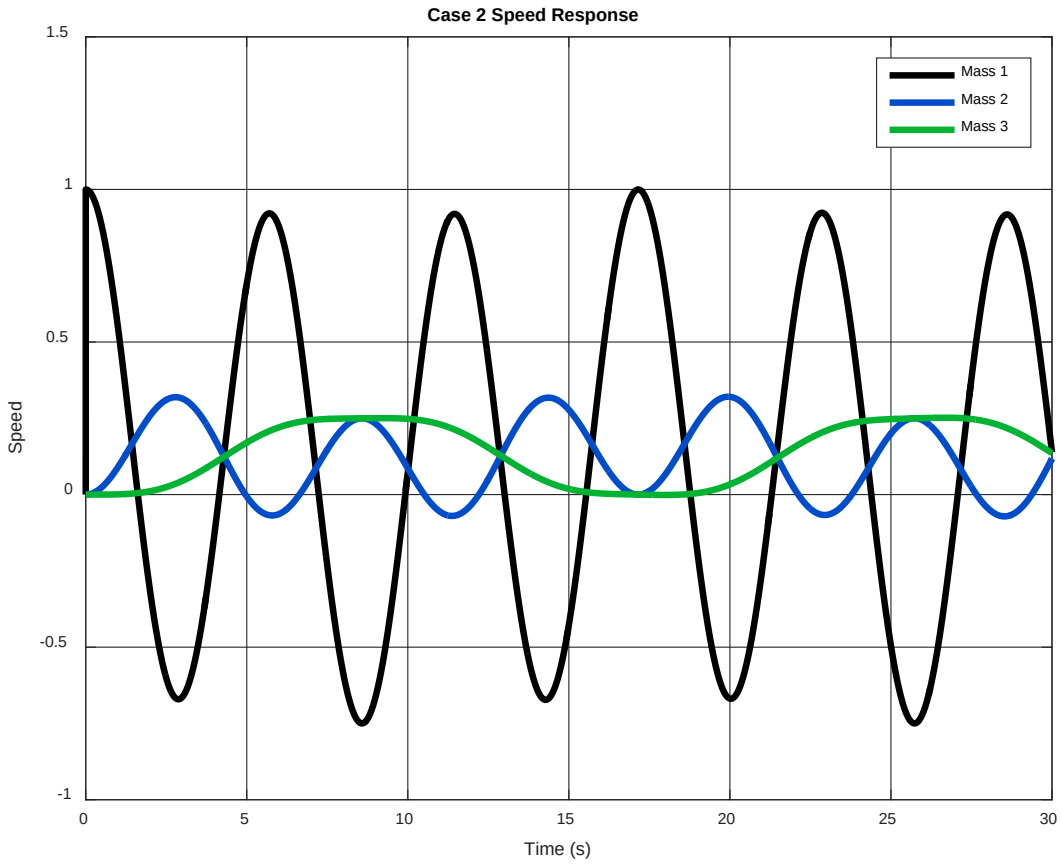

Figure 50. Case 2 Mass Speeds

The results of the simulation show that the first and third mass oscillate against each other with a period of roughly 5.7 seconds. This is clearly the 0.175 Hz mode. The shape of this mode predicted that second mass would oscillate against the first at 20% of the peak-peak amplitude which is almost exactly the difference in peak-peak amplitudes. The 0.175 Hz mode shape also predicted that the third mass would oscillate with the first mass at 2% of the peak-peak amplitude which is visible in the way the oscillations of the third mass flatten out during peaks when the first mass is oscillating the opposite direction. The simulation also shows the third mass is oscillating at a slower frequency that has a period of roughly 17.2 seconds. This is clearly the 0.058 Hz mode which had a mode shape that predicted the first and second mass

would oscillate at 33% and 37% of the amplitude and 180° out of phase of the third mass. This interaction is visible by noting that that peaks of the first and second mass are higher when the third mass peaks low, and their peak lower when the third mass peaks high.

7.2.3. Case 3

This case is identical to Case 2 with the exception of adding friction to the system. The damping coefficients were experimentally chosen to demonstrate a response where one mode is more heavily damped than the other. The coefficients are recorded in Table XII

Table XII: Mass and Spring Constants for Case 3

Variable	Value
m1	1.00
m2	5.00
m3	2.00
k1	1.00
k2	0.20
d0	0.10
d1	0.30
d2	0.01

Note that the friction between the first and second mass is much higher than the friction between the second and third mass. The two oscillatory mode frequencies and damping for this system are shown in Table XIII.

Table XIII: Oscillatory Modes Frequency and Damping for Case 3

Mode (#)	Frequency (Hz)	%Damping
1	0.0578	7.35
2	0.1712	20.30

The first mode has a frequency of 0.058 Hz, so the period of oscillation is expected to be about 17.3 seconds. The second mode has a frequency of 0.171 Hz, so the period of oscillation is expected to be about 5.8 seconds. Because of the added friction the second mode is almost three times more damped than the first. The frequencies of oscillation are also slightly less than they

were in the undamped case. This is because the friction restricts and slows down the motion of the system. The first mode shape is shown in Figure 51 and the second mode shape is shown in Figure 52.

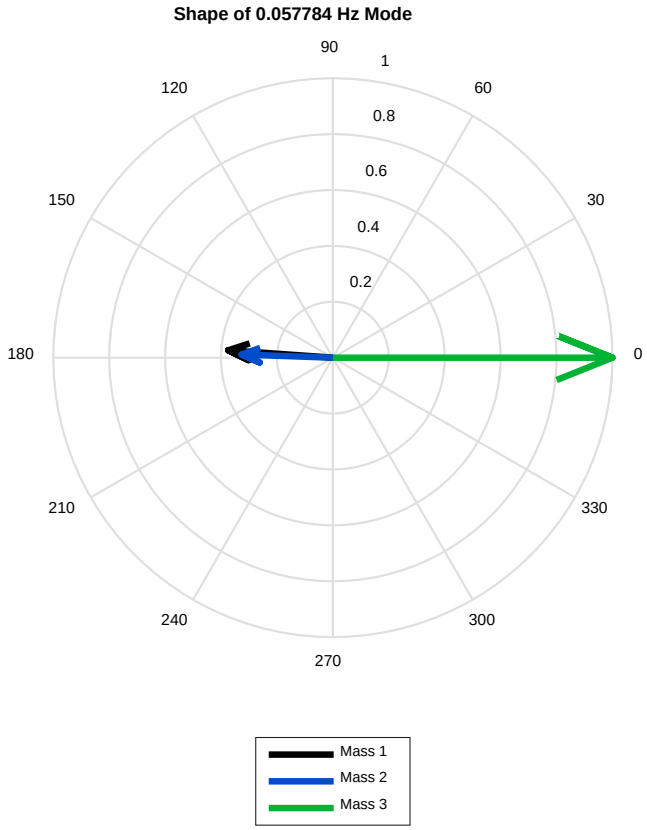

Figure 51. Case 3 Mode Shape for 0.058 Hz Mode

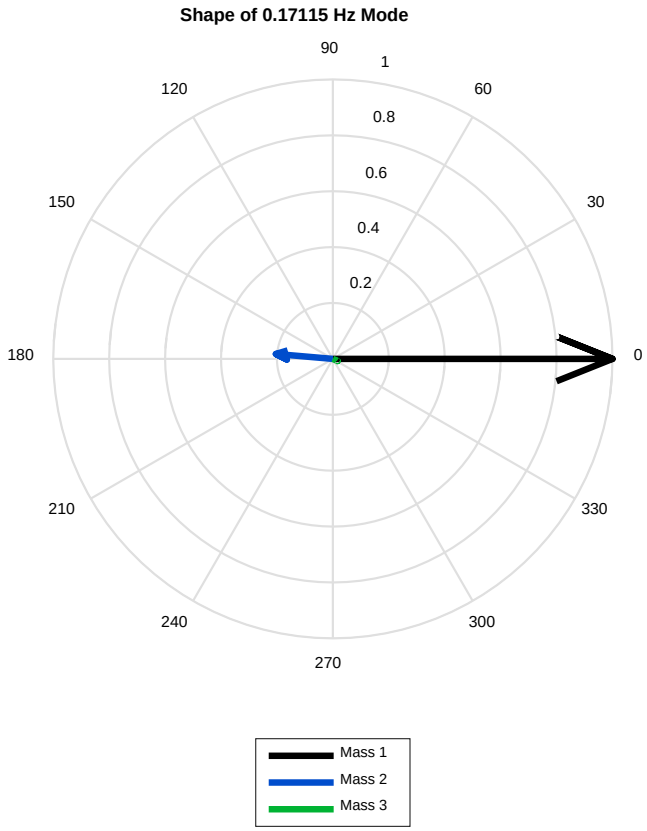

Figure 52. Case 3 Mode Shape for 0.171 Hz Mode

Both mode shapes of Case 3 are nearly the same as the mode shapes of Case 2. The 0.058 Hz mode shape can be interpreted as the third mass oscillating with the largest amplitude about 180° out of phase with the first and second mass, which will have about 37% and 33%, respectively, the peak-peak amplitude of the third mass. The 0.171 Hz mode shape can be interpreted as the first mass oscillating with the largest amplitude in phase with the third mass which has about 1% the peak-peak amplitude. The second mass oscillates about 180° out of phase with first and third mass at about 20% the peak-peak amplitude of the first mass. Note that the masses are no longer oscillating perfectly 180° out of phase with each other due to the

friction. The simulation results for the mass displacements are shown in Figure 53. The mass velocities are shown in Figure 54.

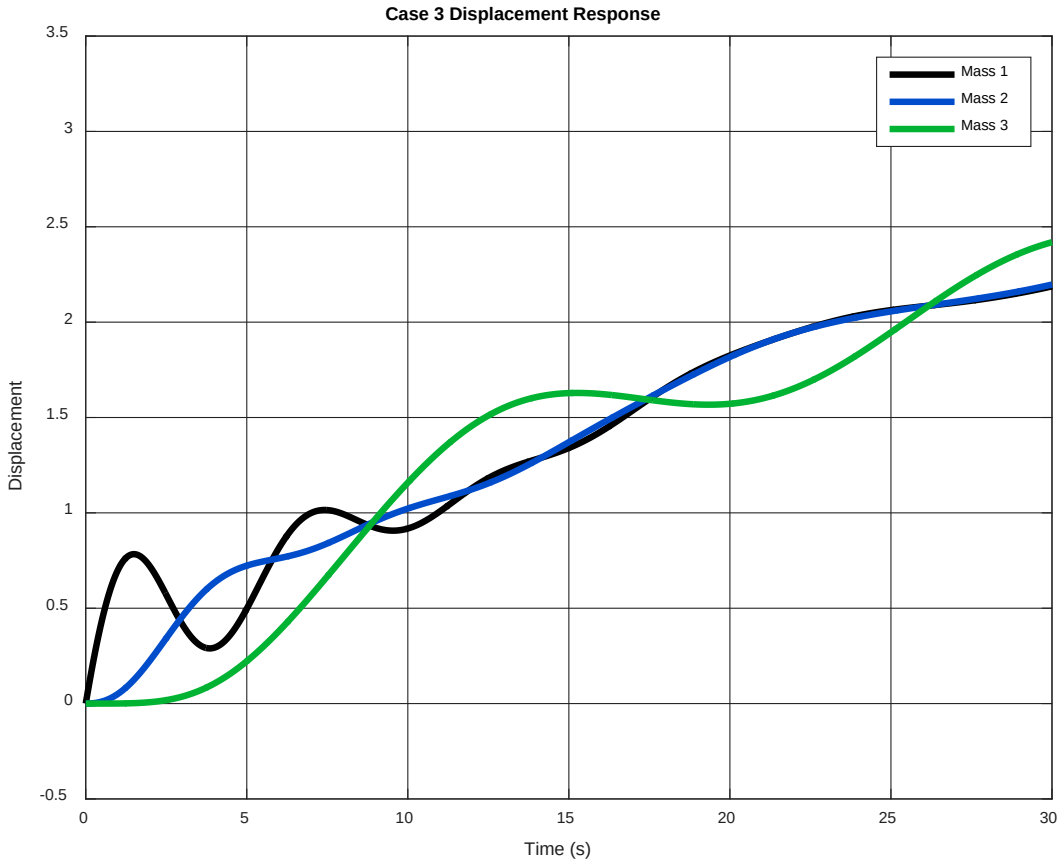

Figure 53. Case 3 Mass Displacement

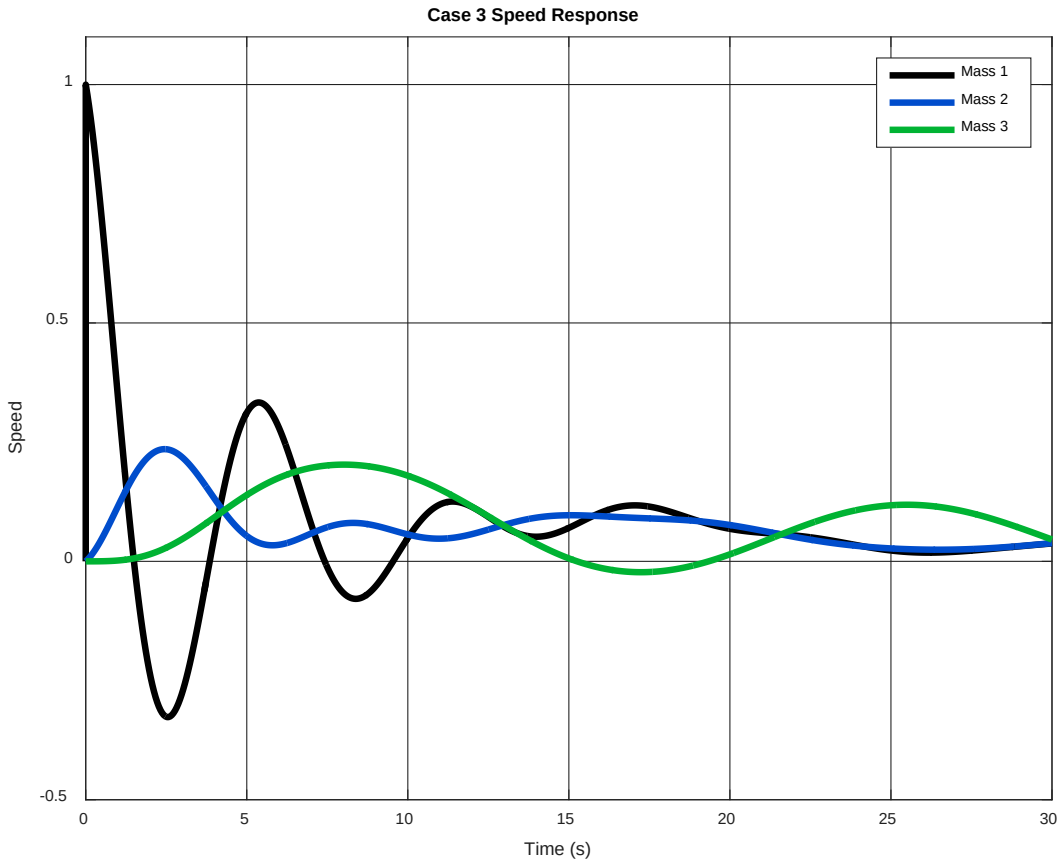

Figure 54. Case 3 Mass Speeds

The results of the simulation show that the first and third mass oscillate against each other with a period of roughly 5.8 seconds. This is the 0.171 Hz mode. The shape of this mode predicted that second mass would oscillate against the first at 20% of the peak-peak amplitude but because of the damping this is hard to see. The 0.171 Hz mode shape also predicted that the third mass would oscillate with the first mass at 1% of the peak-peak amplitude which is not visible due to the damping. The simulation also shows the third mass is oscillating at a slower frequency that has a period of roughly 17.3 seconds. This is the 0.058 Hz mode which had a mode shape that predicted the first and second mass would oscillate at 33% and 37% of the amplitude and 180° out of phase of the third mass. This amplitude is not very clear because of

the damping but the phase is very clear towards the end of simulation. The interesting thing to note about this simulation is that the faster mode damps out much quicker, leaving only the slower mode towards the end of the simulation.

7.3. Eigenanalysis of MicroWECC

As stated before, power systems are inherently non-linear, so before eigenanalysis can be performed, the system must first be linearized around steady-state operating conditions. Power systems also contain controls such as exciters, governors, and power system stabilizers that make eigenanalysis for small systems significantly more complicated than spring-mass-damper systems. Fortunately, PST contains a script that linearizes and calculates the associated eigenvalues and eigenvalues of a given power system.

Calling the PST script *svm_mgen_Batch* takes the load flow and dynamic data from a power system data file and calculates the A, B, C, and D state-space matrices. The A matrix is approximated using perturbations on the system states. Once the A matrix is calculated, the MATLAB function *eig* is used to extract the eigenvalues and right eigenvectors. These are saved as variables to be used for finding the frequency, damping, and shape of the modes. The code used to linearize the model can be found in Appendix S.

Analysis was performed by selecting modes that were with frequencies between 0.1 Hz and 2 Hz and with a damping less than 20%. Mode shapes were developed by normalizing the maximum right eigenvector to an amplitude of one and to an angle of zero. The code used for this analysis can be found in Appendix T. This process was repeated for the MiniWECC and the results were compared. The following sections compare four well-known modes of oscillation between the two models. The results can be found in Appendix U.

7.3.1. North-South A Mode

The NS Mode A is nominally near 0.25 Hz and is the lowest frequency mode found in wNAPS. The damping of this mode is typically anywhere from 10-15% depending on the time of year and amount of loading [12]. In general, the mode shape is described by the northern half of generators oscillating against the southern half of generators. The results from the eigenanalysis performed on the MicroWECC found that the lowest frequency of oscillation occurs at 0.19 Hz with 17% damping. To ensure this was the NS Mode A, a map of the modal shape was created. This was compared to the MiniWECC (version 3C), where the NS Mode A had a frequency of 0.22 Hz with 6% damping. Figure 55 shows the map of the modal shape for the MicroWECC and Figure 56 shows the map of the modal shape for the MiniWECC. Note that circle diameter is proportional to magnitude calculated from the eigenanalysis and red circles represent generators that are approximately 180° out of phase with generators represented by blue circles. All 9 generators of the MicroWECC are represented in the associated map. The corresponding generators in the MiniWECC are shown with additional generators of significant magnitude in the associated map.

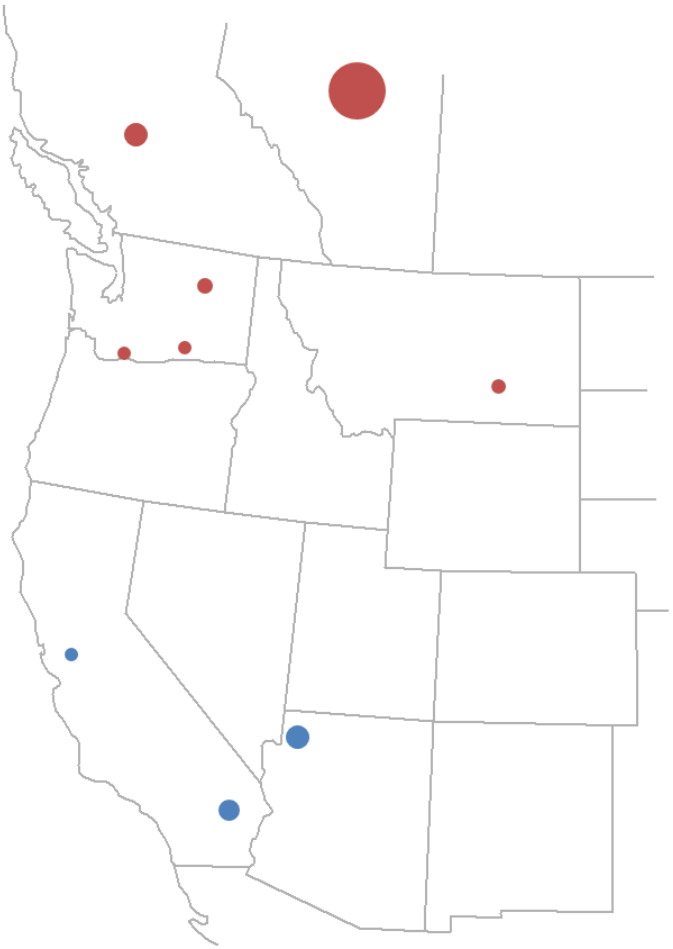

Figure 55. Map of NS Mode A Shape for MicroWECC.
Circle diameter proportional to magnitude. Red oscillates against blue.

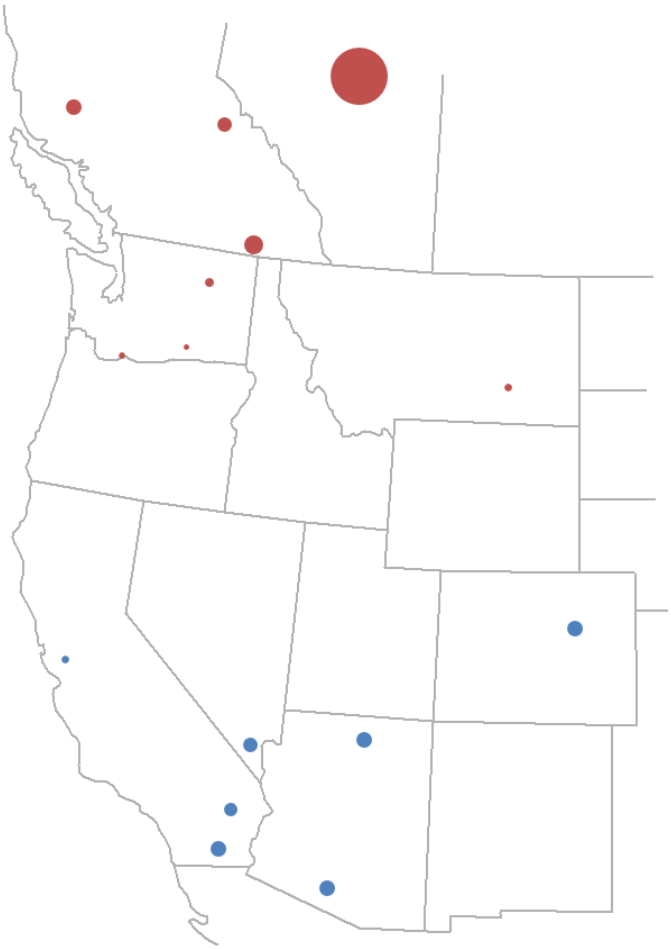

Figure 56. Map of NS Mode A Shape for MiniWECC.
Circle diameter proportional to magnitude. Red oscillates against blue.

The results show that MicroWECC has almost the same mode shape as the MiniWECC for this mode despite having significantly less generators. The largest difference between the models is the damping percentage. The frequency of this mode for both models is below the nominal frequency, with the MicroWECC mode frequency being slightly lower than the MiniWECC. This is most likely due to the three southern generators of the MicroWECC being significantly larger than the corresponding generators in the MiniWECC.

7.3.2. North South Mode B

The NS Mode B is similar to NS Mode A in terms of shape but has a typical frequency of 0.34 Hz to 0.4 Hz and damping of 5% to 10% depending on loading [12]. The modal shape has the Alberta area oscillating against the northern generators which oscillate against the southern generators. Performing eigenanalysis on the MicroWECC revealed the second mode to have a frequency of 0.35 Hz with 4% damping. A map of the modal shape was created and compared to the modal shape of the NS Mode B where the frequency was 0.37 Hz and 3.7% damping. Figure 57 shows the map of the modal shape for the MicroWECC and Figure 58 shows the map of the modal shape for the MiniWECC.

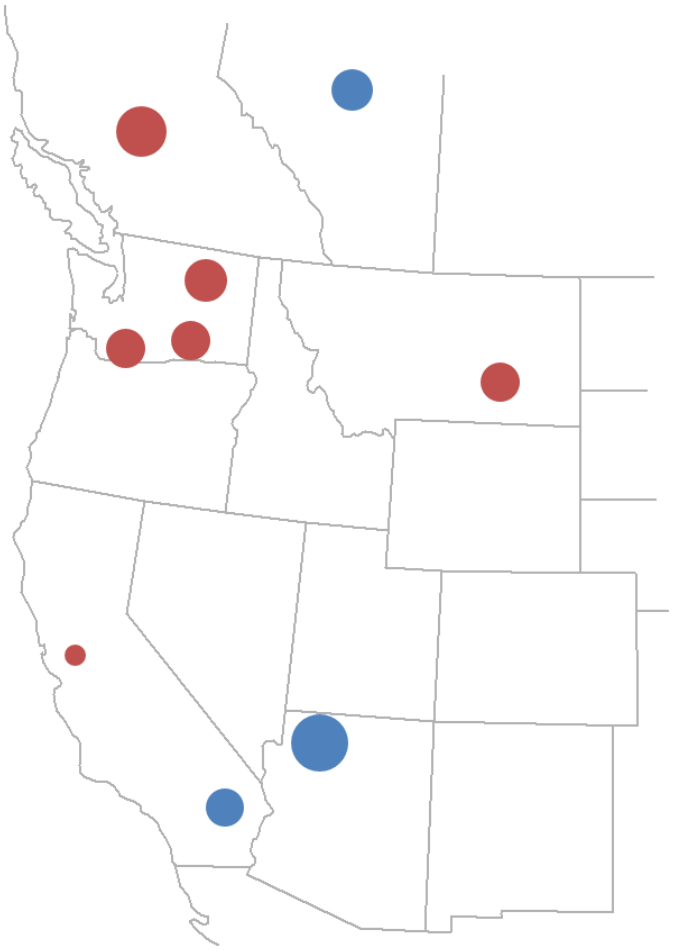

Figure 57. Map of NS Mode B Shape for MicroWECC.
Circle diameter proportional to magnitude. Red oscillates against blue.

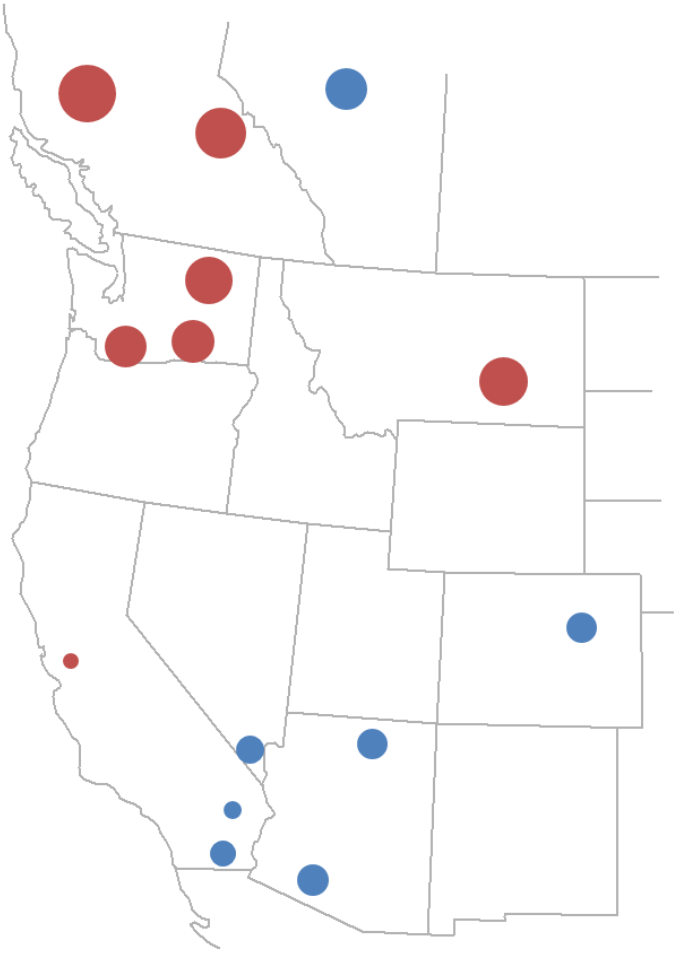

Figure 58. Map of NS Mode B Shape for MiniWECC.
Circle diameter proportional to magnitude. Red oscillates against blue.

The resulting maps show that the mode shapes are very similar between both models. The largest difference is that Gen 9 (Navajo) in the MicroWECC has the largest amplitude, but in the MiniWECC, the Gen 4 (BC) has the largest amplitude. Otherwise, the amplitudes are comparable.

7.3.3. BC Mode

The BC Mode is a fairly well-understood mode of oscillation with a nominal frequency near 0.6 Hz. The mode shape is described as British Colombia oscillating against the rest of the system. Eigenanalysis on the MicroWECC revealed the next mode at 0.51 Hz and 10% damping.

Initially it was thought this was not the BC Mode because in the MiniWECC the BC Mode oscillated at 0.62 Hz and 5.7% damping. A map of the modal shape for both models were created and compared to determine if this mode was the BC mode. Figure 59 shows the map of the modal shape for the MicroWECC and Figure 60 shows the map of the modal shape for the MiniWECC.

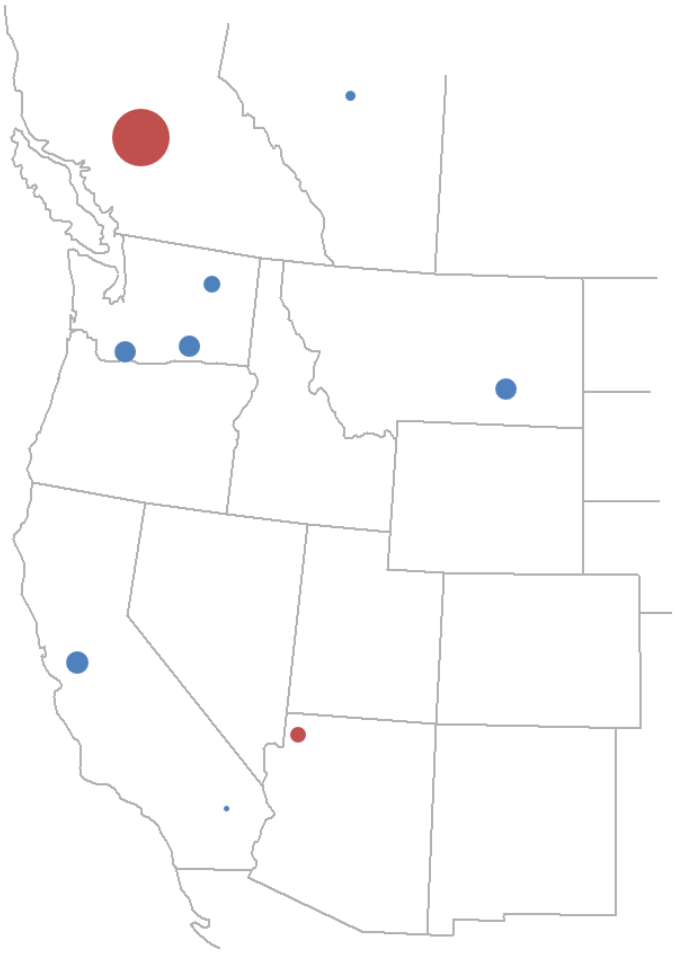

Figure 59. Map of BC Mode Shape for MicroWECC.
Circle diameter proportional to magnitude. Red oscillates against blue.

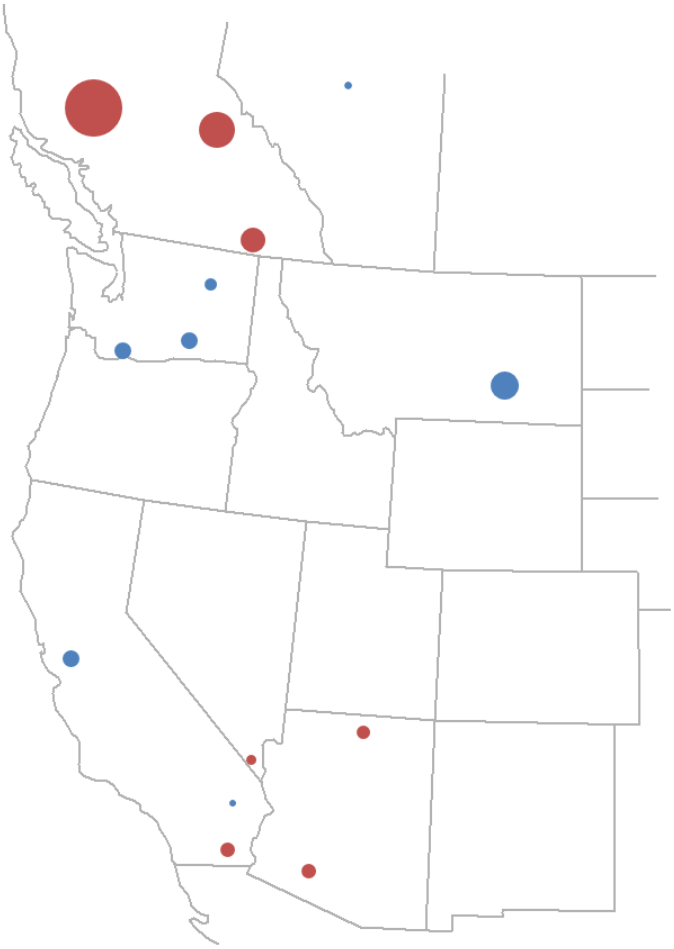

Figure 60. Map of BC Mode Shape for MiniWECC.
Circle diameter proportional to magnitude. Red oscillates against blue.

Both maps reveal that the BC generators have the largest amplitude of oscillation, and generally oscillate against the rest of the system. These shapes confirm it is the BC Mode. The frequency of the BC Mode in the MicroWECC is considerably lower than the nominal frequency for the BC Mode and represents the largest difference in frequency for the four modes that were analyzed. This is likely due to the size of the BC generator in MicroWECC being significantly larger than the northwestern BC generator in the MiniWECC.

7.3.4. Montana Mode

The Montana Mode, sometimes called the Colstrip Mode, has a nominal frequency near 0.8 Hz and typically has damping greater than 10%. The modal shape is described by Colstrip oscillating against the rest of the system. Eigenanalysis revealed the fourth mode of the MicroWECC had a frequency of 0.72 Hz with 11% damping. A map of the modal shape was created and compared to the modal shape of the MiniWECC Montana Mode where the frequency was determined to be 0.69 Hz at 10% damping. Figure 61 shows the map of the modal shape for the MicroWECC and Figure 62 shows the map of the modal shape for the MiniWECC.

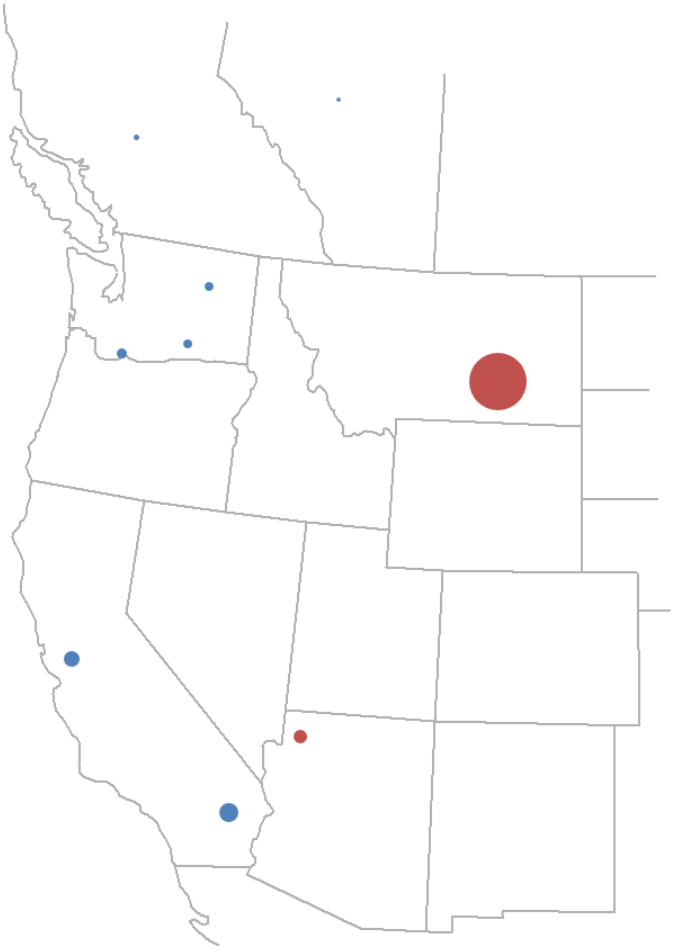

Figure 61. Map of Montana Mode Shape for MicroWECC.
Circle diameter proportional to magnitude. Red oscillates against blue.

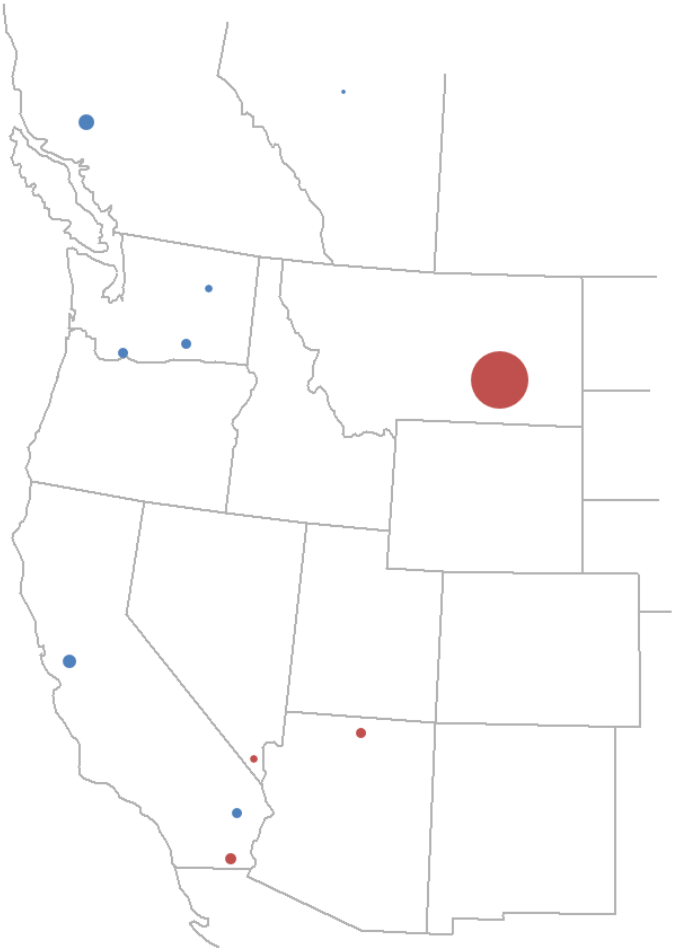

Figure 62. Map of Montana Mode Shape for MiniWECC.
Circle diameter proportional to magnitude. Red oscillates against blue.

Once again, the modal shapes show many similarities confirming the Montana Mode was accurately replicated in the reduced-order model. The frequency and damping of this mode between the two models are also very similar.

8. PSLF MicroWECC Parameters

PSLF (Positive Sequence Load Flow) is commercial software developed and supported by GE's Energy Consulting group that is widely used for planning studies and simulating electric power grids. Like PST, it can be used to perform load flows and transient stability studies. It was considered necessary to build the MicroWECC model in PSLF because the client was more familiar with PSLF and had no direct experience with PST models. This would help assist the client build the MicroWECC model in RSCAD software that interfaces directly to the RTDS.

8.1. Static Data

The first step in building the model in PSLF was to create the static file and add busses, transformers, and transmission line data from the PST model. Resistance and reactance values were copied directly from the PST model. Generation, load, and shunt capacitance values were also added to the static file.

8.2. PST and PSLF Sub-Transient Generator Models

Next, the dynamics file in PSLF was edited to include the equivalent parameters for dynamic models required for transient simulations. Generator models were completed first. The block diagram of PST's sub-transient generator model is shown in Figure 63. The block diagram of PSLF's sub-transient generator model is shown in Figure 64.

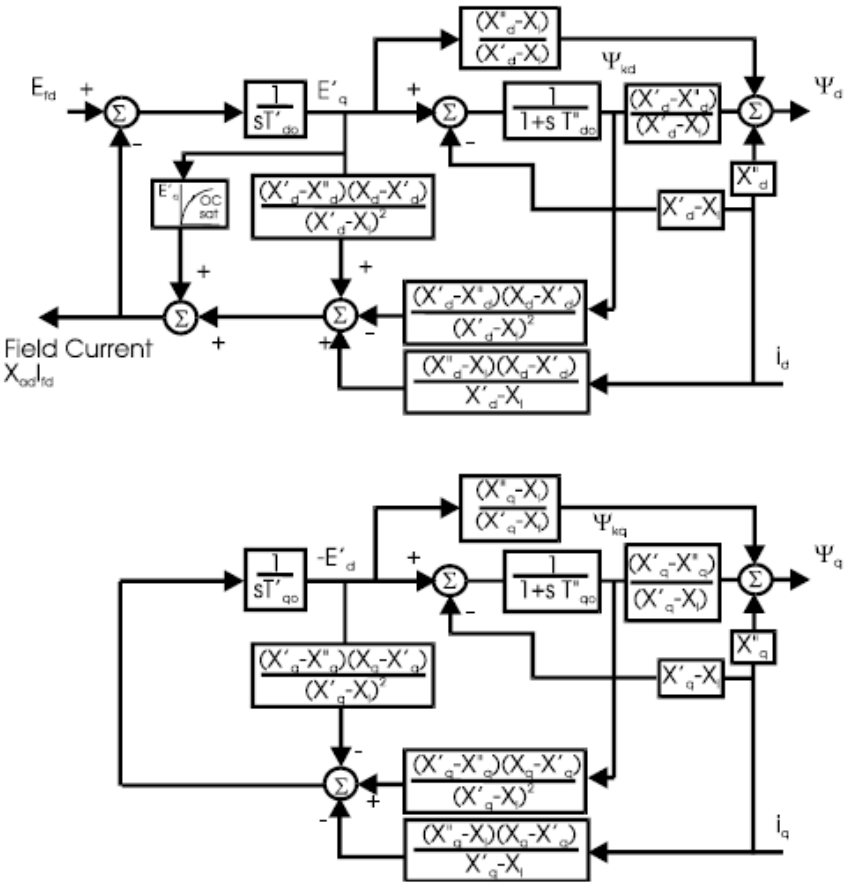

Figure 63. PST Block Diagram of Direct and Quadrature Axes of Sub-transient Generator Model

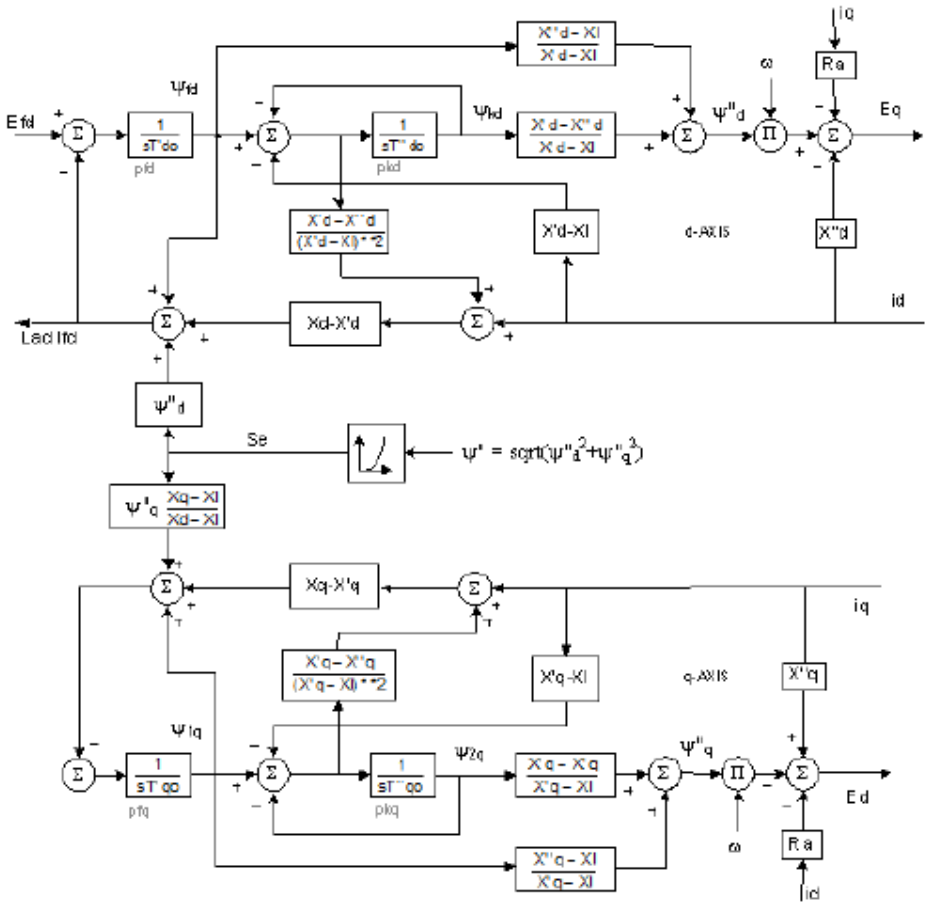

Figure 64. PSLF Block Diagram of Direct and Quadrature Axes of Sub-Transient Generator Model

The generator models have many similarities but are not equivalent. However, they both use the same parameters. The parameter values that were taken from PST and used in the PSLF model are summarized in Table XIV.

Table XIV: Sub-Transient Generator Model Parameters for PST and PSLF

Parameter	PST Variable	PSLF Variable	Value (Steam/Gas Gen)	Value (Hydro Gen)	Unit
Leakage reactance	x_l	*L1*	0.17	0.17	pu
Resistance	r_a	*Ra*	0.00	0.00	pu
D-axis synchronous reactance	x_d	*Ld*	2.00	1.20	pu
D-axis transient reactance	x_d'	*Lpd*	0.20	0.30	pu
D-axis sub-transient reactance	x_d"	*Lppd*	0.18	0.22	pu
D-axis transient time constant	T_{do}'	*Tpdo*	7.50	6.00	sec
D-axis sub-transient time constant	T_{do}"	*Tppdo*	0.025	0.025	sec
Q-axis synchronous reactance	x_q	*Lq*	1.90	0.70	pu
Q-axis transient reactance	x_q'	*Lpq*	0.70	0.23	pu
Q-axis sub-transient reactance	x_q"	*Lppq*	0.18	0.22	pu
Q-axis transient time constant	T_{qo}'	*Tpqo*	0.50	0.06	sec
Q-axis sub-transient time constant	T_{qo}"	*Tppqo*	0.06	0.04	sec
Inertia constant	*H*	*H*	6.50	5.00	sec
Local damping coefficient	d_o	*D*	0.00	0.00	pu
System damping coefficient (not used)	d_1	NA	NA	NA	pu
Saturation factor at 1 pu flux	*S(1.0)*	*S1*	0.05	0.05	pu
Saturation factor at 1.2 pu flux	*S(1.2)*	*S12*	0.30	0.30	pu

8.3. PST and PSLF Exciter Models

Next the exciter models were compared. The PST model "exc_st3," represents an IEEE type ST3 excitation system and is shown in Figure 65. The model used in PSLF, "exst3a," represents a modified IEEE type ST3 excitation system and is shown in Figure 66.

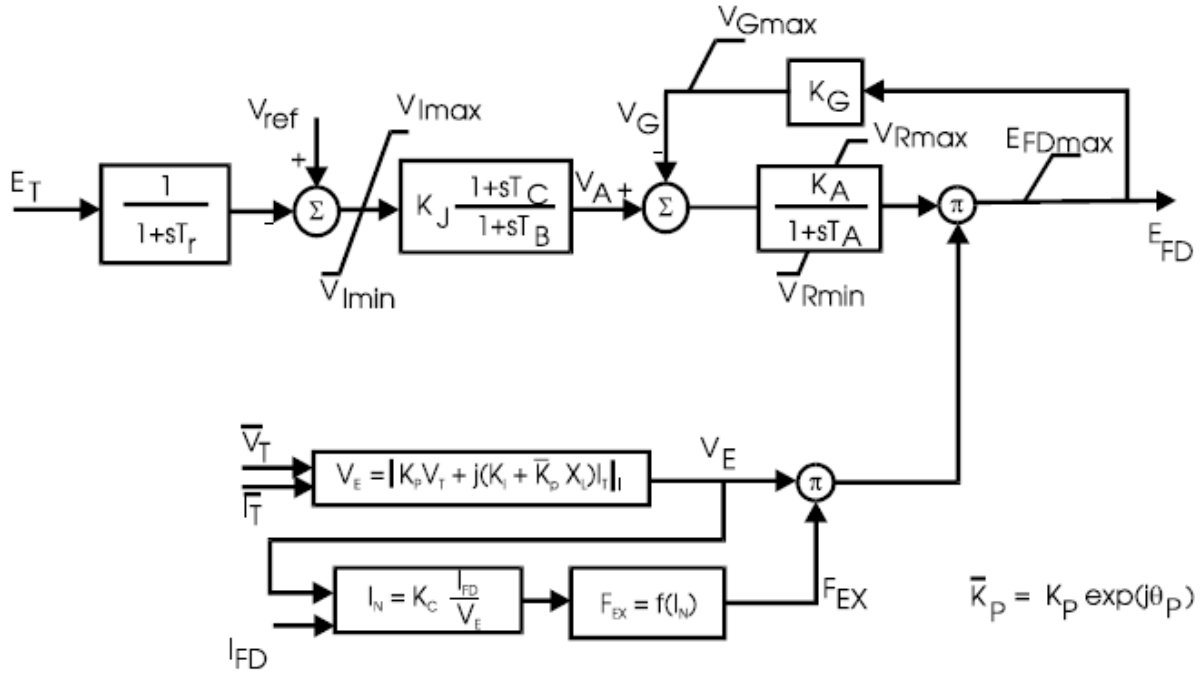

Figure 65. PST ST3 Exciter Model Block Diagram

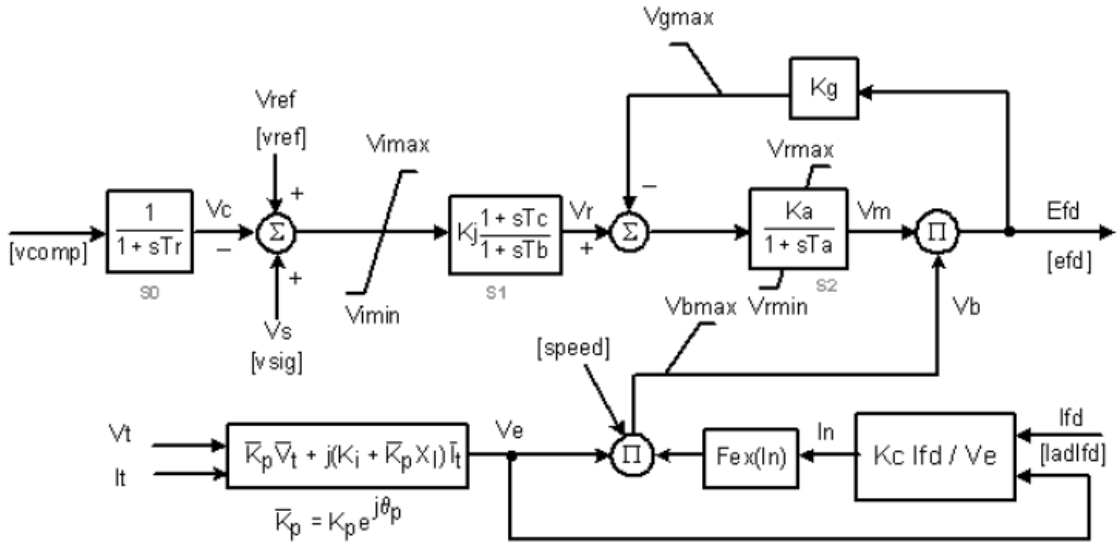

Figure 66. PSLF "esst3a" Model Block Diagram

The exciter models are nearly identical with the exception of the limiter E_{fdmax} in the PST model and the limiter *Vbmax* in the PSLF model. The parameters taken from the PST model and used in the PSLF exciter model are summarized in Table XV.

Table XV: ST3 Exciter Parameters for PST and PSLF

Parameter	PST Variable	PSLF Variable	Value	Unit
Input filter time constant	T_R	*Tr*	0.00	sec
Voltage regulator gain	K_A	*Ka*	200.00	
Voltage regulator time constant	T_A	*Ta*	0.02	sec
Voltage regulator lag time constant	T_B	*Tb*	10.00	sec
Voltage regulator lead time constant	T_C	*Tc*	1.00	sec
Max voltage regulator output	V_{Rmax}	*Vrmax*	5.00	pu
Min voltage regulator output	V_{Rmin}	*Vrmin*	-5.00	pu
Max internal signal	V_{Imax}	*Vimax*	0.10	pu
Min internal signal	V_{Imin}	*Vimin*	-0.10	pu
First state regulator gain	K_J	*Kj*	1.00	
Potential circuit gain coefficient	K_P	*Kp*	1.00	
Potential circuit phase angle	q_p	*Angp*	0.00	degrees
Current circuit gain coefficient	K_I	*Ki*	0.00	
Potential source reactance	X_L	*Xl*	0.00	pu
Rectifier loading factor	K_C	*Kc*	0.00	
Max field voltage	E_{FDmax}	NA	5.00	pu
Max excitation voltage	NA	*Vbmax*	5.00	pu
Inner loop feedback constant	K_G	*Kg*	0.00	
Max inner loop voltage feedback	V_{Gmax}	*Vgmax*	5.00	pu

8.4. PST and PSLF Power System Stabilizer Models

PSS models for PST and PSLF were compared next. The block diagram for the PST PSS model is shown in Figure 67. The PSLF PSS model with the closest equivalent block diagram was the “pss1a” model and is shown in Figure 68.

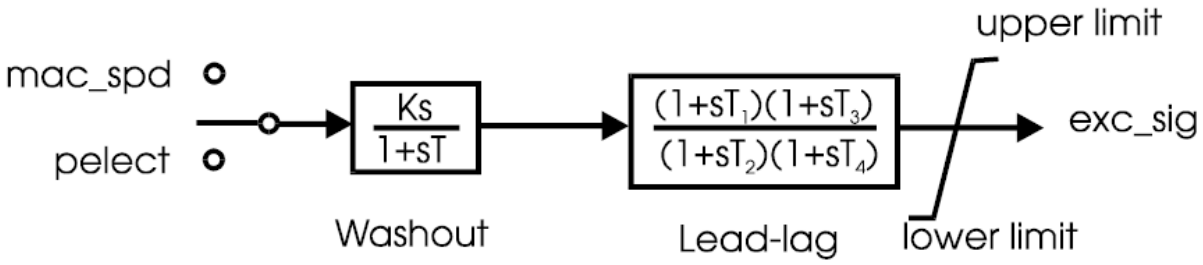

Figure 67. PST PSS Model Block Diagram

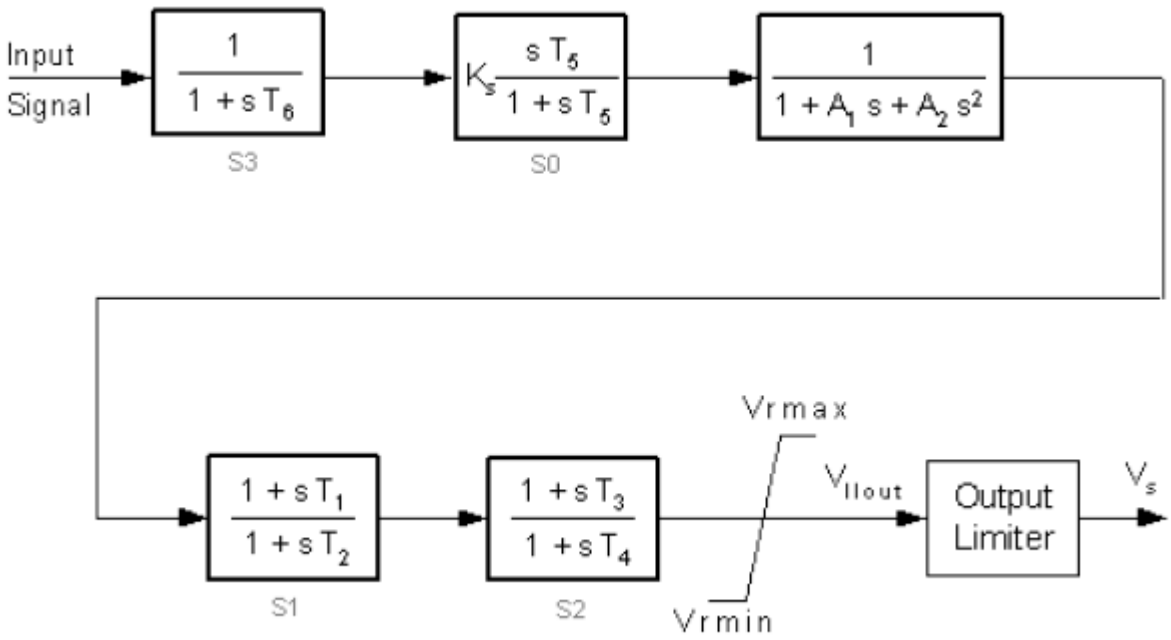

Figure 68. PSLF “pss1a” Model Block Diagram

The main difference between these models is that the PSLF model has extra blocks *S3* and the notch filter after *S0*. Setting the time constant *T6* and the notch filter parameters *A1* and

A2 to zero removes these blocks. The other remaining difference is that in the PSLF model the time constant *T5* is in both the numerator and denominator of *S0*. Setting the K_s in the PSLF model to *K* in the PST model divided by *T5* will ensure both models have the same gain. The PSS parameters for PST and PSLF models are summarized in Table XVI.

Table XVI: PSS Parameters for PST and PSLF

Parameter	PST Variable	PSLF Variable	Value	Unit
Input (1 = speed)	*Input Type*	*j*	1.00	
Gain	*K*	NA	20.0	
Stabilizer Gain	NA	*Ks*	10.0	sec
Washout Time Constant	*T*	*T5*	2.00	sec
Notch Filter Parameter	NA	*A1*	0.00	
Notch Filter Parameter	NA	*A2*	0.00	
Lead Time Constant	T_1	*T1*	0.25	sec
Lag Time Constant	T_2	*T2*	0.04	sec
Lead Time Constant	T_3	*T3*	0.20	sec
Lag Time Constant	T_4	*T4*	0.03	sec
Transducer Time Constant	NA	*T6*	0.00	sec
Max Output Limit	*Upper Limit*	*Vrmax*	0.10	pu
Min Output Limit	*Lower Limit*	*Vrmin*	-0.10	pu

8.5. PST and PSLF Governor Models

The last dynamic models that were compared were turbine-governor models. The block diagram for the PST model is shown in Figure 69. The PSLF model with the closest equivalent block diagram was “tgov1” and is shown in Figure 70.

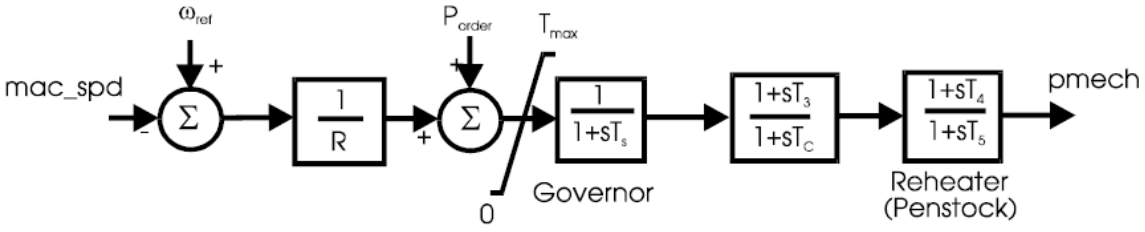

Figure 69. PST Simple Turbine-Governor Model Block Diagram

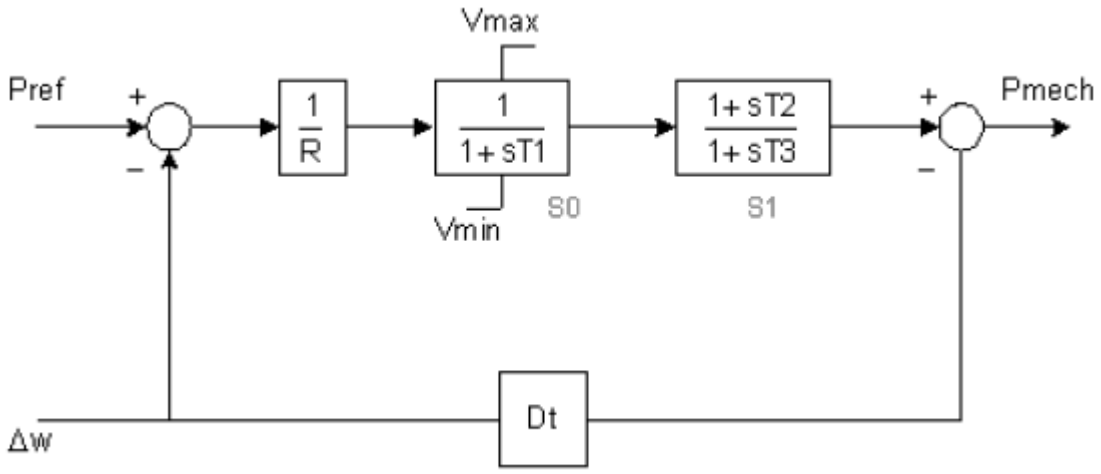

Figure 70. PSLF "tgov1" Model Block Diagram

The differences between the two models are that the PST model has an additional block that models either a reheater or penstock with two time constants, and the PSLF model has damping feedback controlled by the *Dt* block. The damping was set to zero in the PSLF model but there was no way to add the additional time constants to the block diagram. These time constants were near zero for the steam and gas turbine-governors, effectively removing the block, but in the hydro turbine-governors they were non-zero constants used to model the penstock. Therefore, each model was created in Simulink and the time constant *T2* in the PSLF model was experimentally adjusted to the point where step responses between hydro turbine-governor models were as similar as possible. The responses to a step input at t=1s are shown in Figure 71. The MATLAB code and Simulink model used to generate these responses can be found in Appendix V.

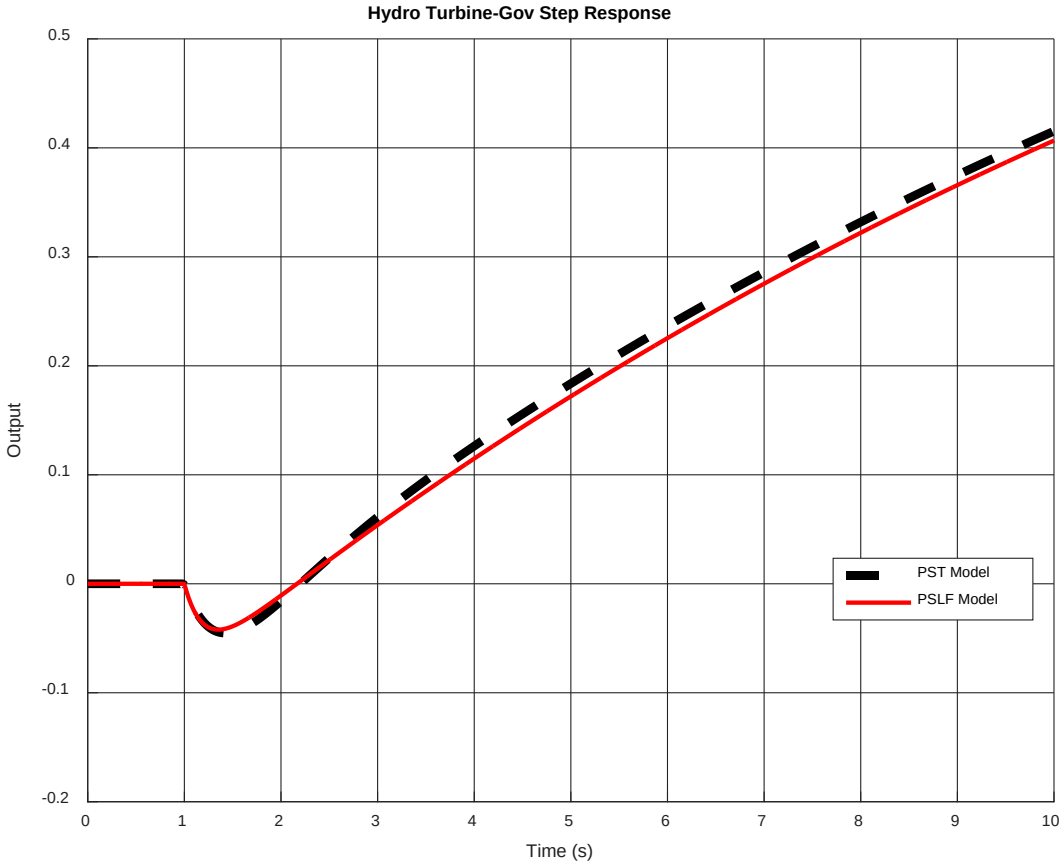

Figure 71. Step Response for PST and PSLF Hydro Turbine-Governors

The parameters for steam, gas, and hydro turbine-governors for PST are shown in Table XVII. PSLF parameters for steam, gas, and hydro turbine-governors are shown in Table XVIII.

Table XVII: PST Turbine-Governor Model Parameters

Parameter	PST Variable	Steam Value	Gas Value	Hydro Value	Unit
Droop	*1/R*	20*	5	20	pu
Max Limit	T_{max}	1.0	1.0	1.0	pu on gen base
Servo Time Constant	T_S	0.1	0.1	0.5	sec
HP Turbine Time Constant	T_C	10	10	15	sec
Transient Gain Time Constant	T_3	3.0	4.0	1.0	sec
HP Turbine Ratio Time Constant	T_4	0.0	0.0	-1.0	sec
Reheater Time Constant	T_5	0.01	0.01	0.5	sec

*For the Gen 2 governor this value is 5. For Gen 9 governor this value is 4.

Table XVIII: PSLF Turbine-Governor Model Parameters

Parameter	PST Variable	Steam Value	Gas Value	Hydro Value	Unit
Droop	*R*	.05*	.20	.05	pu
Max Valve Position	*Vmax*	1.0	1.0	1.0	pu on gen base
Min Valve Position	*Vmin*	0.0	0.0	0.0	sec
Steam Bowl Time Constant	*T1*	0.1	0.1	0.2	sec
Numerator Time Constant of S1 Block	*T2*	3.0	4.0	-1.0	sec
Reheater Time Constant	*T3*	10	10	15	sec
Turbine Damping Coefficient	*Dt*	0.0	0.0	0.0	pu

*For the Gen 2 governor this value is 0.2. For Gen 9 governor this value is 0.25.

8.6. PSLF MicroWECC Validation

The turbine-governor information concluded the dynamic data section of the PSLF file. The MicroWECC model in PSLF was then compared to the model in PST by conducting a transient simulation. Note that for this comparison the first version of the MicroWECC was used, however this does not diminish the results, because the point was to show that a PST model can be built in PSLF using the parameters and methods provided above to attain very similar simulation results. For this simulation, the line connecting bus 3 and bus 4 was disconnected at t=1s. The speeds of generators were then compared. Speeds for generators 1-3 are shown in Figure 72. Speeds for generators 4-6 are shown in Figure 73. Speeds for generators 7-9 are shown in Figure 74.

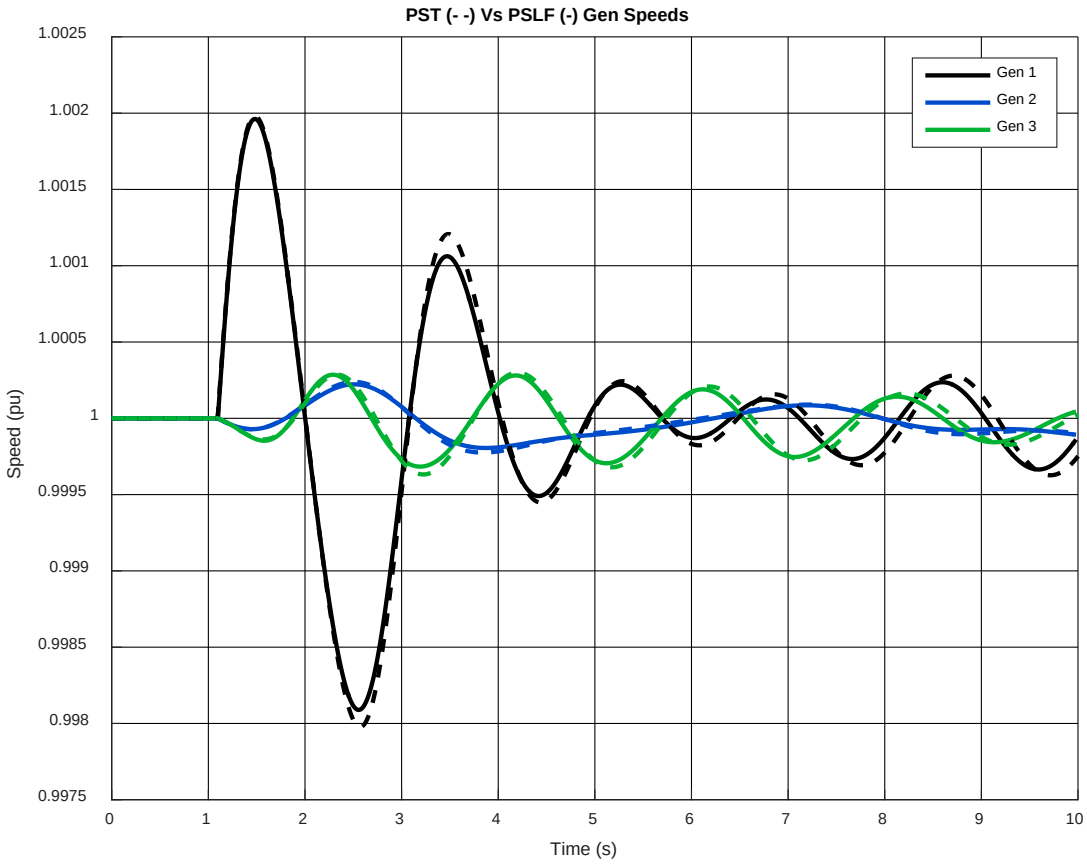

Figure 72. Comparison of PST and PSLF Generators 1-3 Speeds

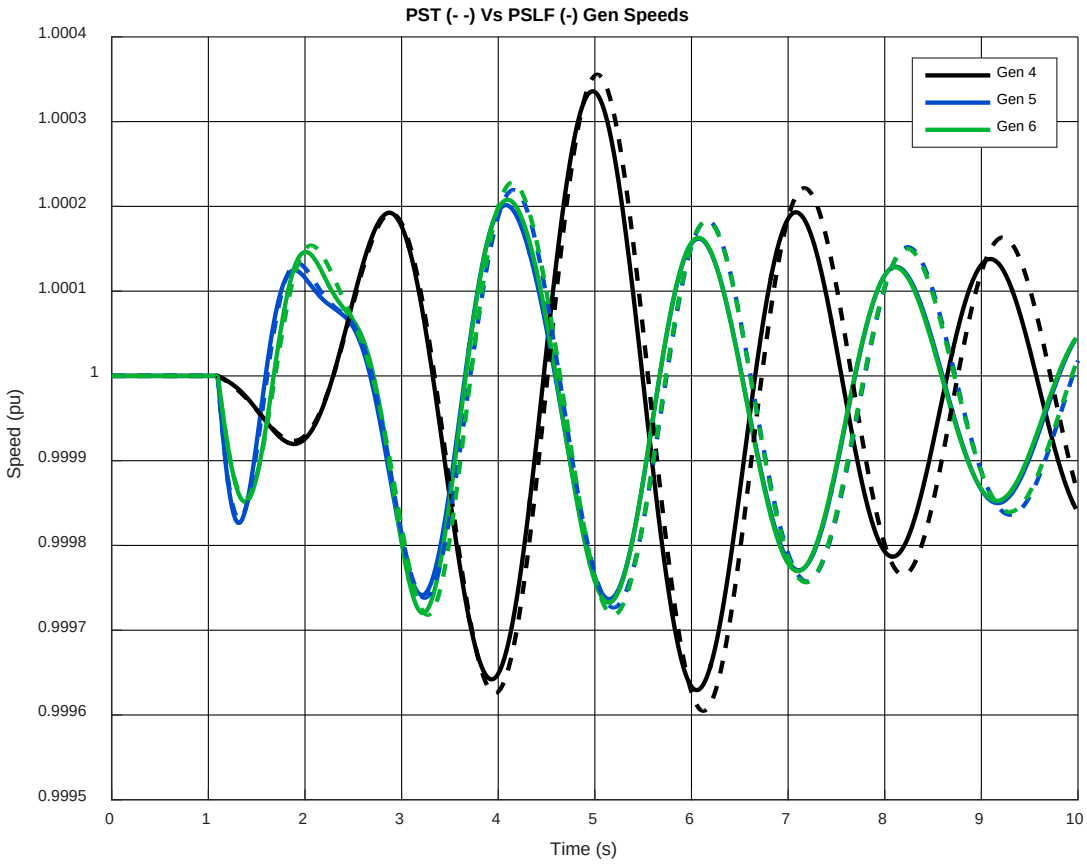

Figure 73. Comparison of PST and PSLF Generators 4-6 Speeds

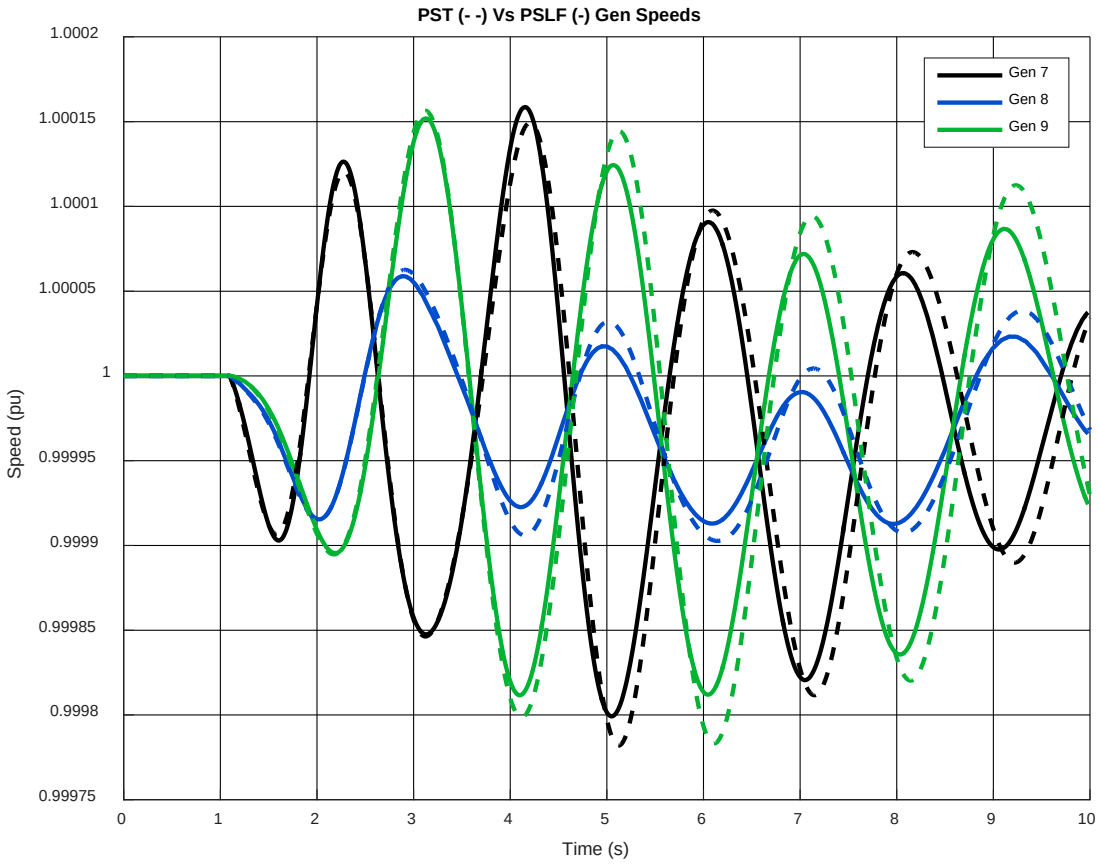

Figure 74. Comparison of PST and PSLF Generators 7-9 Speeds

Note that PST generator speeds are represented by dotted lines and PSLF generator speeds are represented by solid lines. From the results, both models are fairly similar but the PSLF generator speeds have slightly more damping than the PST generator speeds. This difference is likely caused by the difference in generator models but is not large enough to cause significant differences in transient simulation results.

9. PSCAD MicroWECC Parameters

PSCAD (Power System Computer-Aided Design) is an electromagnetic transients program used to model and simulate power systems. PSCAD simulates three-phase voltages and currents in contrast to PSLF and PST where voltages and currents are represented by a complex number. PSCAD is the most similar program to RSCAD, so it was decided that suggesting models and parameter values for PSCAD would reduce the amount of work on the client's end when building the MicroWECC in the RTDS.

9.1. Generator Model

The PSCAD model used to represent a generator is the synchronous machine model. A block diagram could not be found in the PSCAD documentation but Figure 75 shows the synchronous machine model connected to a governor, turbine, exciter, and PSS.

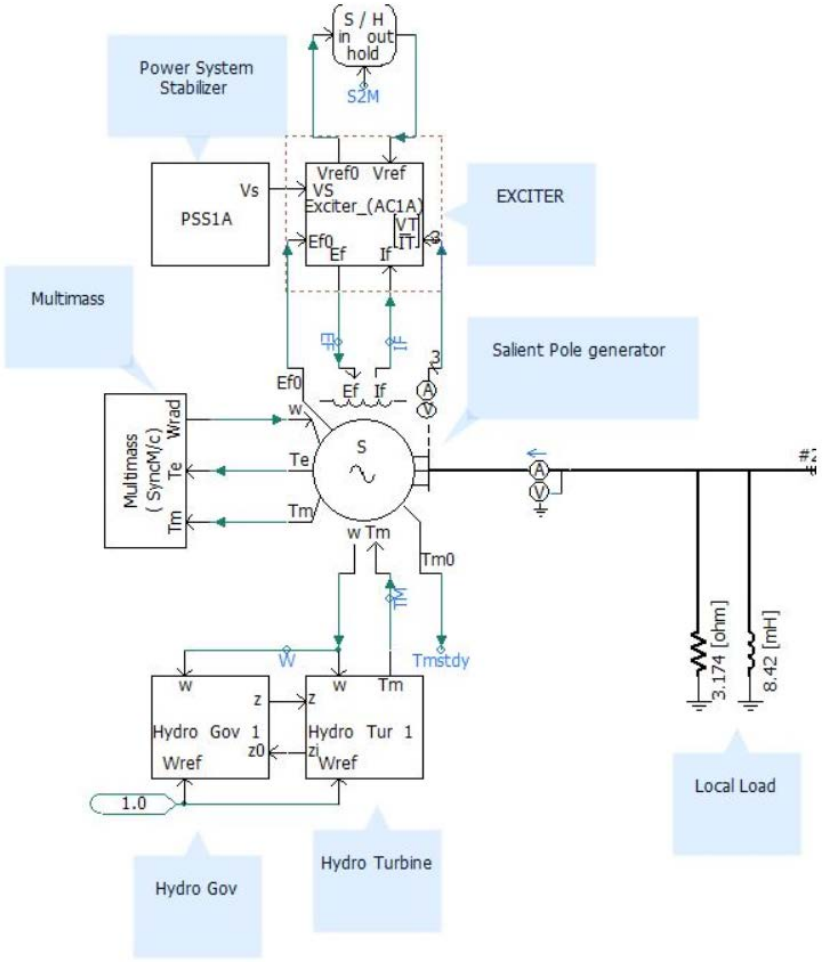

Figure 75. PSCAD Synchronous Machine Model

The generator data format for the model indicated that most of the parameters were identical so the recommended values are summarized in Table XIX.

Table XIX: PSLF Synchronous Machine Model Parameters

PSCAD Parameter	PSCAD Variable	Value (Steam/Gas Gen)	Value (Hydro Gen)	Unit	Note
Rated RMS L-N voltage	NA	11.547	11.547	kV	
Rated RMS Line current	NA	MBase/(20*sqrt(3))	MBase/(20*sqrt(3))	kA	MBase is the power base of the generator
Base angular frequency	NA	376.99	376.99	rad/s	
Inertia constant [H]	*H*	6.50	5.00	sec	
Mechanical friction and windage	*D*	0.00	0.00	pu	
Armature resistance [Ra]	*R*	0.00	0.00	pu	
Potier reactance [Xp]	*L1*	0.17	0.17	pu	X_L = Xp*Air Gap Factor
D-axis synchronous reactance [Xd]	*Ld*	2.00	1.20	pu	
D-axis transient reactance [Xd_]	*Lpd*	0.20	0.30	pu	
D-axis sub-transient reactance [Xd__]	*Lppd*	0.18	0.22	pu	
D-axis transient time constant [Tdo_]	*Tpdo*	7.50	6.00	sec	
D-axis sub-transient time constant [Tdo__]	*Tppdo*	0.025	0.025	sec	
Q-axis synchronous reactance [Xq]	*Lq*	1.90	0.70	pu	
Q-axis transient reactance [Xq_]	*Lpq*	0.70	0.23	pu	
Q-axis sub-transient reactance [Xq__]	*Lppq*	0.18	0.22	pu	
Q-axis transient time constant [Tqo_]	*Tpqo*	0.50	0.06	sec	
Q-axis sub-transient time constant [Tqo__]	*Tppqo*	0.06	0.04	sec	
Air Gap Factor	NA	1.00	1.00		

9.2. PSLF Exciter Model

The exciter model that was recommended for PSCAD is called “ST3A” and represents an IEEE standard model. The block diagram is shown in Figure 76.

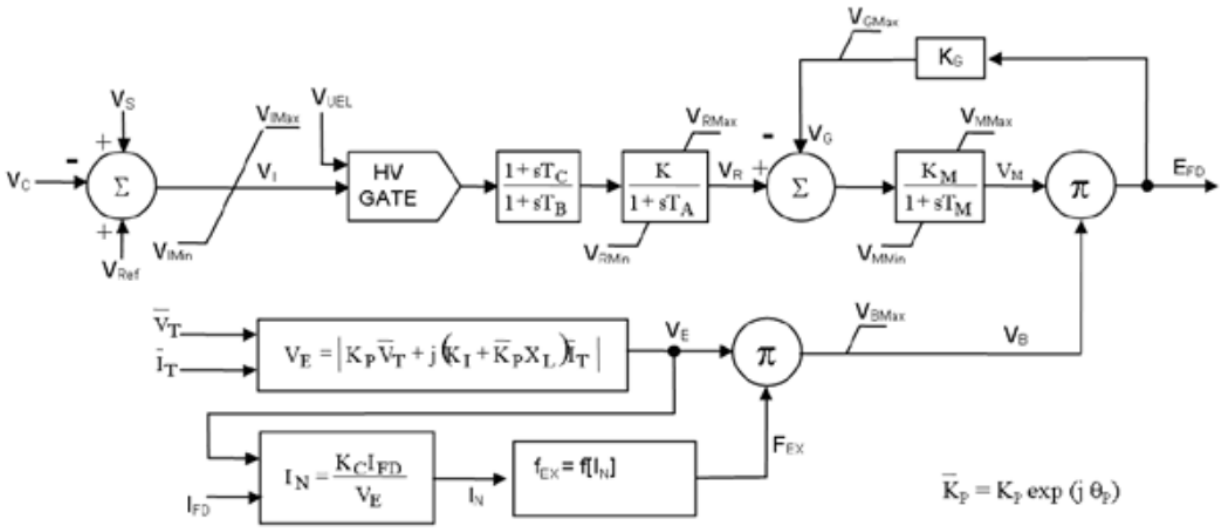

Figure 76. PSCAD "ST3A" Exciter Model Block Diagram

This model is very similar to the models used in PST and PSLF. The main difference is that this model contains an extra block, but this can be bypassed by setting the associated gain to one and the associated time constant to zero. The suggested parameters are shown in Table XX. PSLF variables are provided to clarify where values were derived from.

Table XX: ST3A Exciter Parameters for PSCAD

Parameter	PSCAD Variable	PSLF Variable	Value	Unit
Max field voltage	E_{FDmax}	NA	5.00	pu
Regulator gain	K_A*	*Kj*	1.00	
Rectifier loading factor	K_C	*Kc*	0.00	
Inner loop feedback constant	K_G	*Kg*	0.00	
Current circuit gain coefficient	K_I	*Ki*	0.00	
Potential circuit gain coefficient	K_P	*Kp*	1.00	
Field inner loop gain constant	K_M	*Ka*	200.0	
Regulator time constant	T_A	NA	0.0	sec
Voltage regulator lag time constant	T_B	*Tb*	10.00	sec
Voltage regulator lead time constant	T_C	*Tc*	1.00	sec
Angle of circuit voltage multiplier	*THETA_p*	*Angp*	0.00	degrees
Field inner loop time constant	T_M	*Ta*	0.02	sec
Max excitation voltage	V_{BMAX}	*Vbmax*	5.00	pu
Max inner loop voltage feedback	V_{GMax}	*Vgmax*	5.00	pu
Max error signal	V_{IMax}	*Vimax*	0.10	pu
Min error signal	V_{IMin}	*Vimin*	-0.10	pu
Max voltage regulator output	V_{RMax}	*Vrmax*	5.00	pu
Min voltage regulator output	V_{RMin}	*Vrmin*	-5.00	pu
Max field inner loop	V_{MMax}	*Vrmax*	5.00	pu
Min field inner loop	V_{MMin}	*Vrmin*	-5.00	pu
Potential source reactance	X_L	*Xl*	0.00	pu

*Labeled as K in Figure 76

9.3. PSLF PSS Model

The PSS model that was recommended for PSCAD was the “PSS1A” which represents an IEEE standard model. The block diagram is shown in Figure 77.

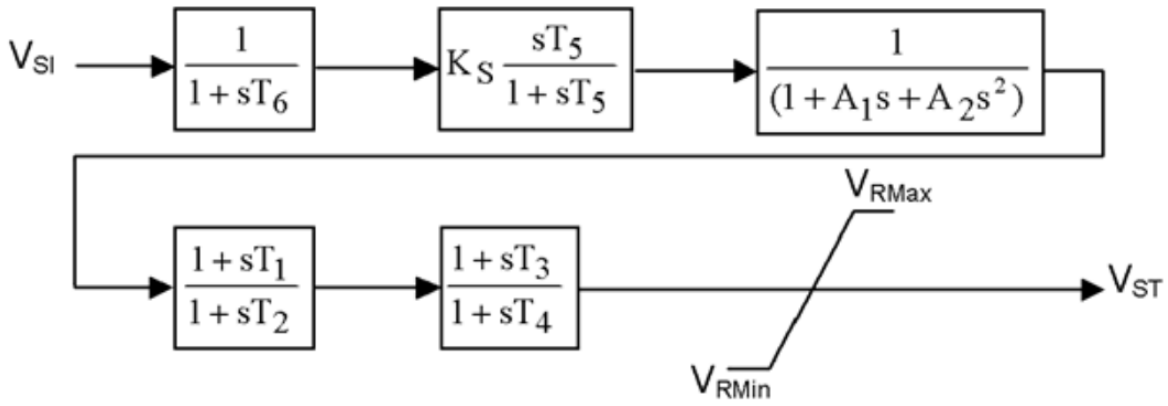

Figure 77. PSCAD "PSS1A" PSS Model Block Diagram

This model is identical to the PSS model used in PSLF. The suggested parameters are equivalent to the PSLF values and are shown in Table XXI.

Table XXI: PSS Parameters for PSCAD

Parameter	PSCAD Variable	Value	Unit
Input Signal		Speed	
PSS Gain	K_S	10.0	
Transducer Time Constant	T_6	0.00	sec
Washout Time Constant	T_5	2.00	sec
Filter Constant	A_1	0.00	
Filter Constant	A_2	0.00	
First Lead Time Constant	T_1	0.25	sec
First Lag Time Constant	T_2	0.04	sec
Second Lead Time Constant	T_3	0.20	sec
Second Lag Time Constant	T_4	0.03	sec
Max Output Limit	V_{RMax}	0.10	pu
Min Output Limit	V_{RMin}	-0.10	pu

9.4. PSCAD Turbine and Governor Models

PSCAD does not have a turbine-governor model that is similar to the simple model used in PSLF. Instead, separate models are used to represent both the turbine and the governor. Turbine and governor models are also separated into thermal and hydro models. The thermal governor model that was recommended for steam and gas governors was "GOV1" and is shown

in Figure 78. The thermal turbine model recommended for steam and gas turbines was “TUR1” and is shown in Figure 79.

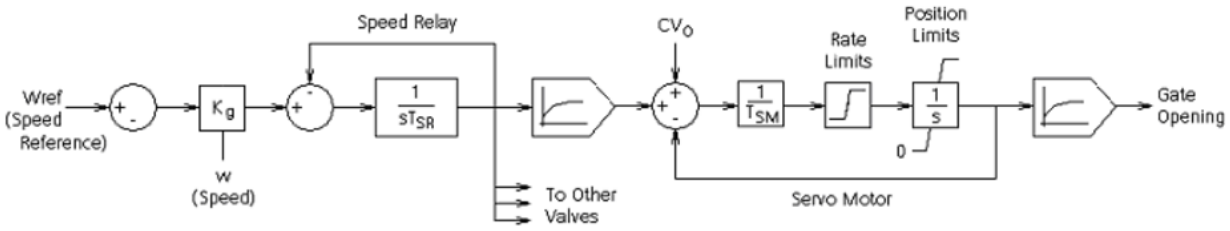

Figure 78. PSCAD “GOV1” Thermal Governor Model Block Diagram

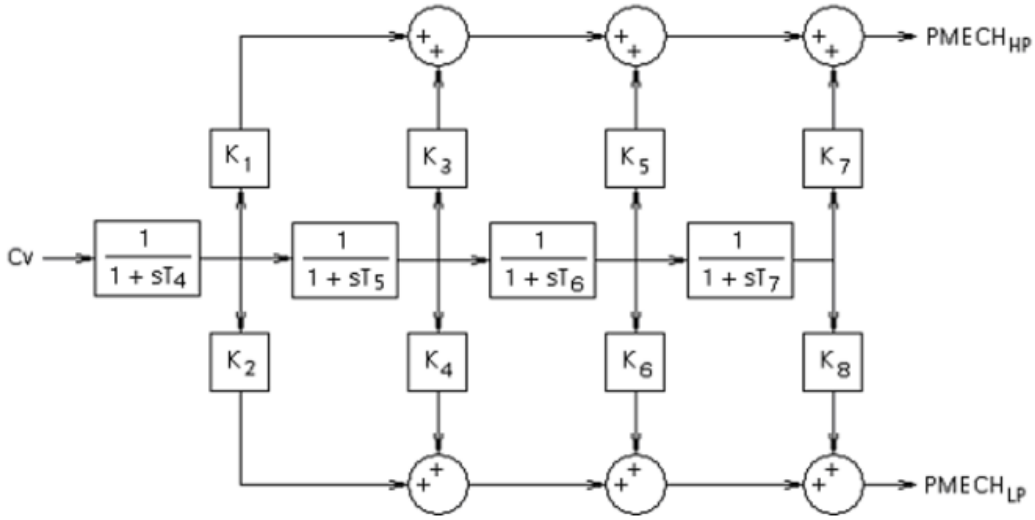

Figure 79. PSCAD “TUR1” Thermal Turbine Model Block Diagram

The hydro governor model recommended was “GOV2” and is shown in Figure 80. The hydro turbine model recommended was “TUR1” and is shown in Figure 81.

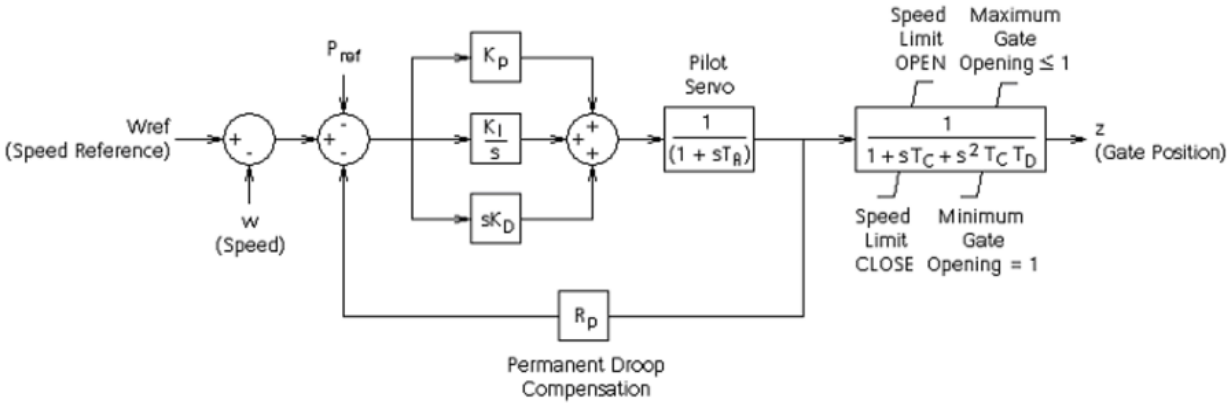

Figure 80. PSCAD "GOV2" Hydro Governor Model Block Diagram

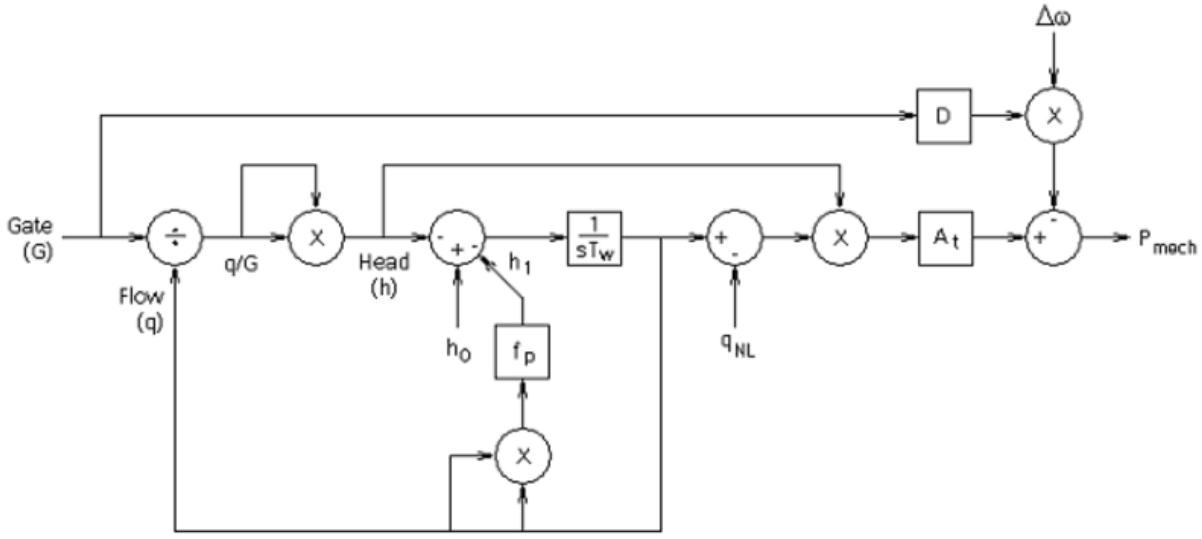

Figure 81. PSCAD "TUR1" Hydro Turbine Model Block Diagram

A significant amount of work went into deriving parameters for both sets of governors and turbines to match the PSLF model. In the end, however, the client decided to implement the PSLF model due to its simplicity. This was done using a summation block, two transfer function blocks, and a limiter. This implementation is shown in Figure 82.

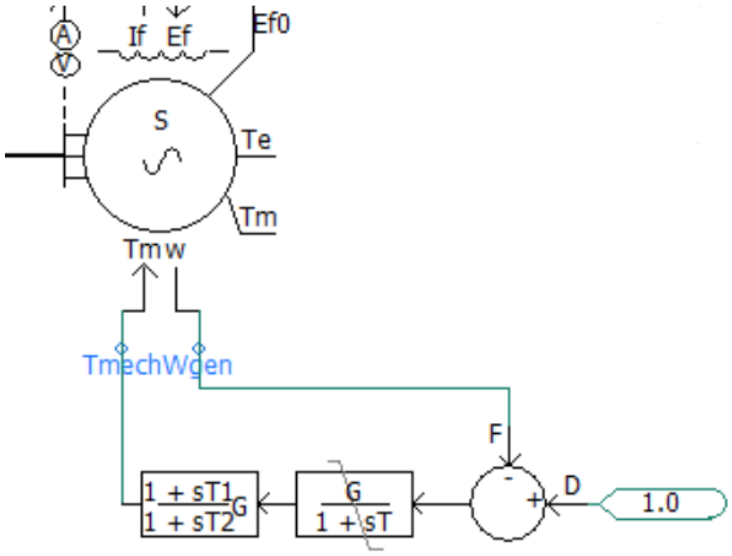

Figure 82. PSCAD Implementation of PSLF Turbine-Governor Model

It was noted that this implementation differs from the PSLF model in that it lacks a mechanical power input. Additionally, the second gain *G* should be removed because it is not needed, or at the very least changed to a different name. However, this was an initial version and the implementation has likely changed to account for these differences. The values that should be used for this model are identical to those used in the PSLF model and are shown in Table XXII

Table XXII: PSCAD Turbine-Governor Model Parameters

Parameter	PSCAD Variable	Steam Value	Gas Value	Hydro Value	Unit
Droop	*G*	20*	5.0	20	pu
Max Limit	*Max*	1.0	1.0	1.0	pu on gen base
Min Limit	*Min*	0.0	0.0	0.0	sec
Steam Bowl Time Constant	*T*	0.1	0.1	0.2	sec
Numerator Time Constant	*T1*	3.0	4.0	-1.0	sec
Denominator Time Constant	*T2*	10	10	15	sec

* For the Alberta governor this value is 5. For the Navajo governor this value is 4.

9.5. Conclusion

The MicroWECC was designed to have approximate impedance, generation, and mode characteristics as the MiniWECC. The MiniWECC had a total of 34 generators, 19 loads, and

122 busses and 2 DC lines. This was reduced to 9 generators, 7 loads, and 1 DC line in the MicroWECC using generator and transmission line equivalencing. A type of modal analysis known as eigenanalysis was then performed on the MiniWECC and MicroWECC models to verify the accuracy of the new model. It was determined that four well-known modes were accurately reproduced; the NS Mode A, NS Mode B, BC Mode, and Montana Mode. The mode shapes were very consistent between models, while the largest differences were that the damping in the NS Mode A was about 10% higher and the frequency of the BC mode was about 0.1 Hz lower in the MicroWECC. Parameters were successfully found to transfer the MicroWECC PST model to PSLF. Despite having different generator models, transient simulation results indicated only minor differences between the two programs. Finally, PSCAD models and parameters were recommended to the client to minimize the work required to build the model in the RTDS simulator. Future work would involve verifying the RTDS simulations against PST or PSLF simulations.

References Cited

[1] About NERC. (2017). Retrieved from https://www.nerc.com/AboutNERC/Pages/default.aspx

[2] About WECC. (n.d.). Retrieved from https://www.wecc.biz/Pages/AboutWECC.aspx

[3] Western Electricity Coordinating Council - Relay Work Group, “Remedial Action Scheme Design Guide,” Dec 2015.

[4] C. A. Stigers, C. S. Woods, J. R. Smith and R. D. Setterstrom, “The acceleration trend relay for generator stabilization at Colstrip,” in *IEEE Transactions on Power Delivery*, vol. 12, no. 3, pp. 1074-1081, Jul 1997. doi: 10.1109/61.636872

[5] D. N. Kosterev, J. Esztergalyos and C. A. Stigers, “Feasibility study of using synchronized phasor measurements for generator dropping controls in the Colstrip System,” in *IEEE Transactions on Power Systems*, vol. 13, no. 3, pp. 755-761, Aug 1998. doi: 10.1109/59.708579

[6] D. Trudnowski, D. Kosterev, J. Undrill, “PDCI Damping Control Analysis for the Western North American Power System,” *Proceedings of the IEEE Power & Energy Society General Meeting*, July 2013.

[7] E. Bahr, “Acceleration Trend Relay Model Specification,” June 2015.

[8] J. Bélanger, P. Venne, and J.-N. Paquin, ‘‘The what, where and why of real-time simulation,’’ in *Proc. PES General Meeting*, Oct. 2010, pp. 37–49.

[9] M. D. Omar Faruque et al., "Real-Time Simulation Technologies for Power Systems Design, Testing, and Analysis," in *IEEE Power and Energy Technology Systems Journal*, vol. 2, no. 2, pp. 63-73, June 2015. doi: 10.1109/JPETS.2015.2427370

[10] D. J. Trudnowski, "Estimating Electromechanical Mode Shape from Synchrophasor Measurements," in *IEEE Transactions on Power Systems*, vol. 23, no. 3, pp. 1188-1195, Aug. 2008. doi: 10.1109/TPWRS.2008.922226

[11] G. Rogers, *Power System Oscillations*. Boston, MA: Kluwer, 2000.

[12] D. J. Trudnowski “Properties of the Dominant Inter-Area Modes in the WECC Interconnect” Jan 2012

10. Appendix A: Continuous to Discrete-Time Filter Script

```
%Discrete-Time Filter Script
%
%Purpose: Convert continuous-time filters from the ATR unit calculations
%to 60-sps discrete time filters using Tustin bilinear method
%
%Author: RJ Hallett
%Date: Feb 2017
%%
clear variables;
close all;

CF1 = tf(1,[.03333 1]);                 %Continuous-time filter 1
DF1 = c2d(CF1,1/60,'tustin');           %Discrete-time filter 1

CF2 = tf([0.7825 1],[.2695 1.174 1]);   %Continuous-time filter 2
DF2 = c2d(CF2,1/60,'tustin');           %Discrete-time filter 2

CF3 = tf([597.7 1],[4.04 1]);           %Continuous-time filter 3
DF3 = c2d(CF3,1/60,'tustin');           %Discrete-time filter 3

CF4 = tf([.001562 0],[.02353 1]);       %Continuous-time filter 4
DF4 = c2d(CF4,1/60,'tustin');           %Discrete-time filter 4

CF5 = tf(250,[1.7065 1]);               %Continuous-time filter 5
DF5 = c2d(CF5,1/60,'tustin');           %Discrete-time filter 5

%% Create Bode plot of each filter for comparison
bode(CF1, DF1)
legend('Continuous','Discrete')
title('Filter 1')

figure
bode(CF2, DF2)
legend('Continuous','Discrete')
title('Filter 2')

figure
bode(CF3, DF3)
legend('Continuous','Discrete')
title('Filter 3')

figure
bode(CF4, DF4)
legend('Continuous','Discrete')
title('Filter 4')

figure
bode(CF5, DF5)
legend('Continuous','Discrete')
title('Filter 5')
```

11. Appendix B: Continuous-Time vs. Discrete-Time Filter Bode Plots

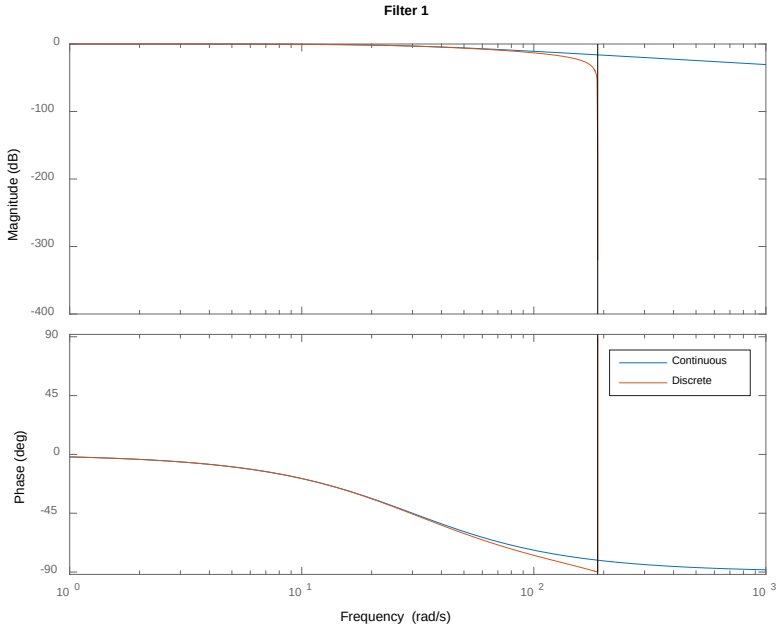

Figure 83. Bode Plot Comparison of Continuous-Time Filter 1 and Resulting Discrete-Time Filter 1

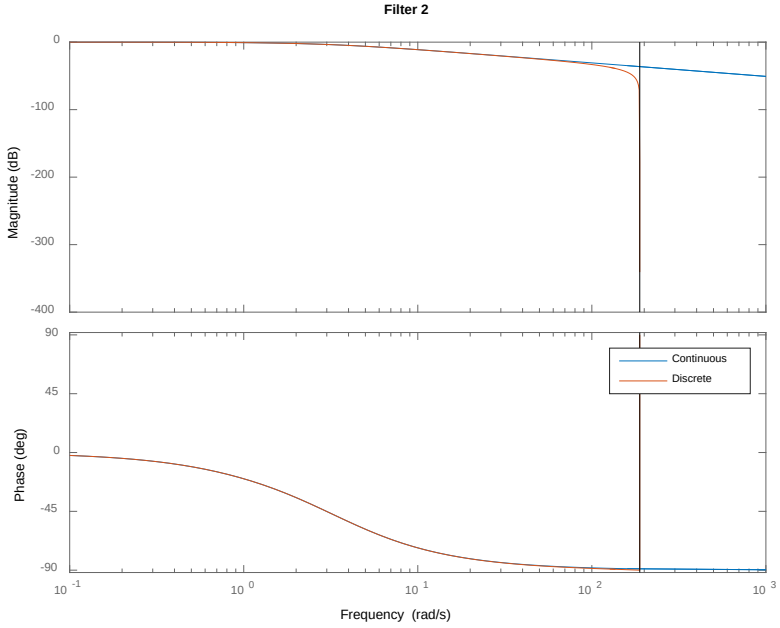

Figure 84. Bode Plot Comparison of Continuous-Time Filter 2 and Resulting Discrete-Time Filter 2

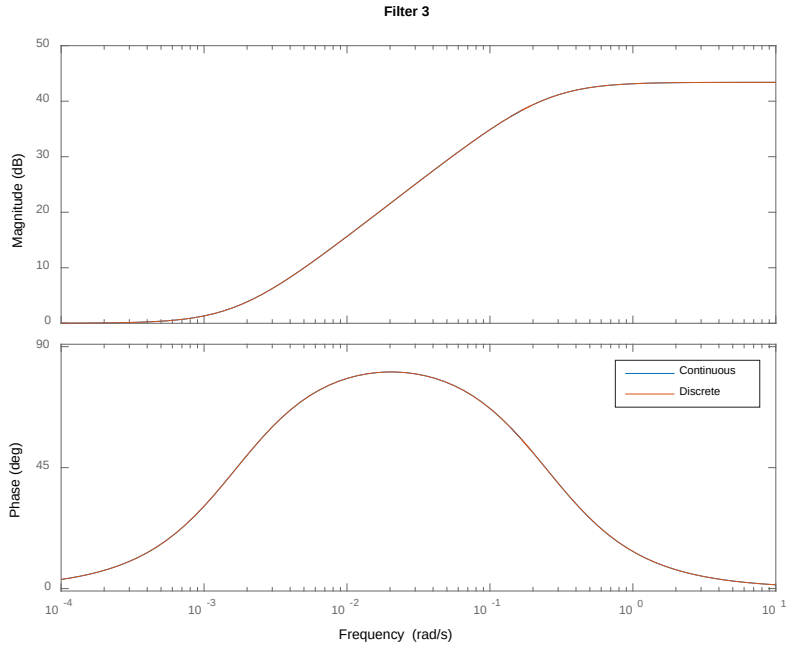

Figure 85. Bode Plot Comparison of Continuous-Time Filter 3 and Resulting Discrete-Time Filter 3

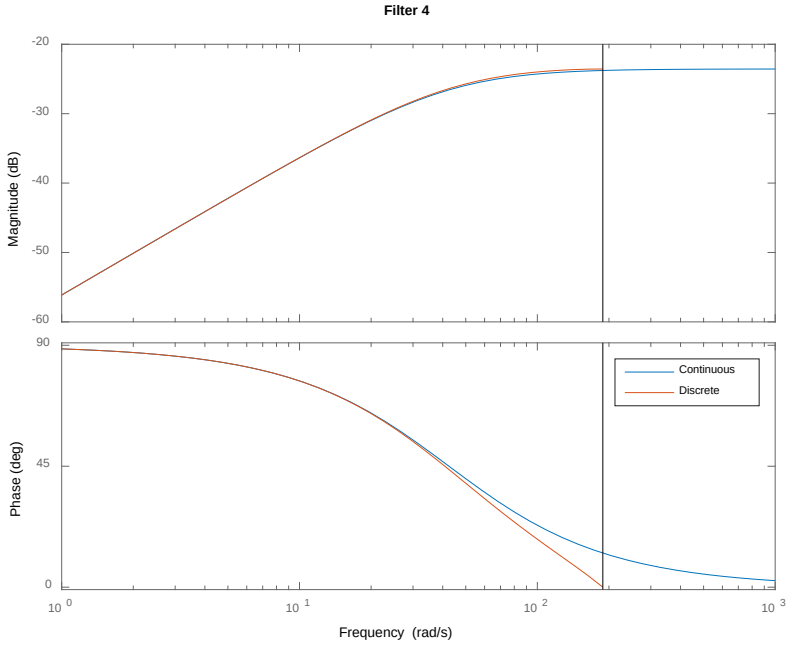

Figure 86. Bode Plot Comparison of Continuous-Time Filter 4 and Resulting Discrete-Time Filter 4

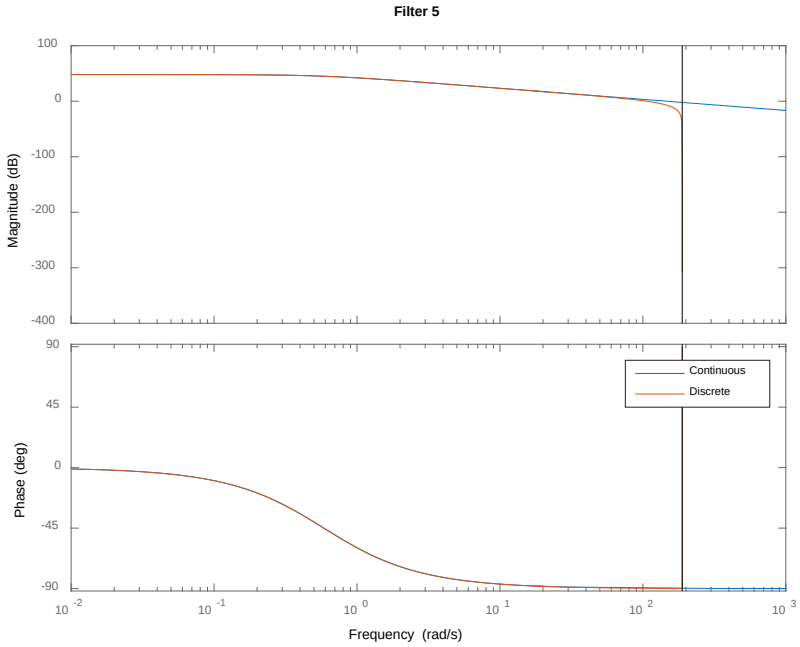

Figure 87. Bode Plot Comparison of Continuous-Time Filter 5 and Resulting Discrete-Time Filter 5

12. Appendix C: Filtering Function Script

```
function mac_trip_states =
funFiltMacTrip(Num,Den,mac_trip_states,RowIn,RowOut,k)
Num = Num./Den(1);
Den = Den./Den(1);

for i=2:length(Den)
    n = k-i+1;
    if n>0
        mac_trip_states(RowOut,k) = -mac_trip_states(RowOut,n)*Den(i) +
mac_trip_states(RowOut,k);
    else
        break
    end
end

for i=1:length(Num)
    n = k-i+1;
    if n>0
        mac_trip_states(RowOut,k) = mac_trip_states(RowIn,n)*Num(i) +
mac_trip_states(RowOut,k);
    else
        break
    end
end
```

13. Appendix D: Machine Trip Logic for ATR

```
function [tripOut,mac_trip_states] =
mac_trip_logic(tripStatus,mac_trip_states,t,kT)
% Purpose: Model the ATR and trip generators.
%
% Inputs:
%   tripStatus = n_mac x 1 bool vector of current trip status.  If
%       tripStatus(n) is true, then the generator corresponding to the nth
%       row of mac_con is already tripped.  Else, it is false.
%   mac_trip_states = storage matrix defined by user.
%   t = vector of simulation time (sec.).
%   kT = current integer time (sample).  Corresponds to t(kT)
%
% Output:
%   tripOut = n_mac x 1 bool vector of desired trips.  If
%       tripOut(n)==1, then the generator corresponding to the nth
%       row of mac_con is will be tripped.  Note that each element of
%       tripOut must be either 0 or 1.
%   mac_trip_states = storage matrix defined by user.

% Version 1.3
% Authors:   Dan Trudnowski, RJ Hallett
% Date:   March 2017

%% Define global variables
global mac_spd %Gen pu speeds; mac_spd(n,kT) = gen n speed at time t(kT)
global pelect  %gen pu powers; pelect(n,kT) = gen n real power at time t(kT)
global n_mac

%% Initialize
tripOut = false(n_mac,1);   %Set tripOut to zero's, length = # of mach's
% Initialize states
if kT==1
    mac_trip_states = zeros(65,length(t)); %Initialize size of storage var
    mac_trip_states(1,1) = 0; %time of last analysis (sample)
    mac_trip_states(2,1) = 0; %time of last analysis (sec.)
    %states for gen. 14 pelect filter
    mac_trip_states(3,1) = pelect(14,1); %gen 14 initial real power (pu)
    mac_trip_states(4,1) = pelect(14,1); %gen 14 initial real power (pu)
    %states for gen. 35 pelect filter
    mac_trip_states(8,1) = pelect(35,1); %gen 35 initial real power (pu)
    mac_trip_states(9,1) = pelect(35,1); %gen 35 initial real power (pu)
    %states for gen. 36 pelect filter
    mac_trip_states(13,1) = pelect(36,1); %gen 36 initial real power (pu)
    mac_trip_states(14,1) = pelect(36,1); %gen 36 initial real power (pu)
    %states for gen. 37 pelect filter
    mac_trip_states(18,1) = pelect(37,1); %gen 37 initial real power (pu)
    mac_trip_states(19,1) = pelect(37,1); %gen 37 initial real power (pu)
    %states for gen. 14 mac_spd filter
    mac_trip_states(5,1) = mac_spd(14,1); %gen 14 initial speed (pu)
    mac_trip_states(6,1) = mac_spd(14,1); %gen 14 initial speed (pu)
    %states for gen. 35 mac_spd filter
    mac_trip_states(10,1) = mac_spd(35,1); %gen 35 initial speed (pu)
    mac_trip_states(11,1) = mac_spd(35,1); %gen 35 initial speed (pu)
```

```
    %states for gen. 36 mac_spd filter
    mac_trip_states(15,1) = mac_spd(36,1); %gen 36 initial speed (pu)
    mac_trip_states(16,1) = mac_spd(36,1); %gen 36 initial speed (pu)
    %states for gen. 37 mac_spd filter
    mac_trip_states(20,1) = mac_spd(37,1); %gen 37 initial speed (pu)
    mac_trip_states(21,1) = mac_spd(37,1); %gen 37 initial speed (pu)

    onlineStatus = bin2dec('1111'); %Initialize online status
    mac_trip_states(65,1) = onlineStatus;   %Save online status
end

%% Start ATR Algorithm and Update Generator States
if (t(kT)-mac_trip_states(2,1)) >= (1/60 - 1/600) %Ensure 60 sps sample rate
    mac_trip_states(1,1) = mac_trip_states(1,1) + 1; %update analysis sample
    k = mac_trip_states(1,1) + 1; %increment current filter index
    mac_trip_states(2,1) = t(kT); %update analysis time

    %Update gen 14 states
    mac_trip_states(4,k) = pelect(14,kT);
    mac_trip_states(6,k) = mac_spd(14,kT);

    %Update gen 35 states
    mac_trip_states(9,k) = pelect(35,kT);
    mac_trip_states(11,k) = mac_spd(35,kT);

    %Update gen 36 states
    mac_trip_states(14,k) = pelect(36,kT);
    mac_trip_states(16,k) = mac_spd(36,kT);

    %Update gen 37 states
    mac_trip_states(19,k) = pelect(37,kT);
    mac_trip_states(21,k) = mac_spd(37,kT);

    %Update online status
    onlineStatus = mac_trip_states(65,1);

%% Unit Calculations
    if k>2
        %Unit Gains
        UnitSpeedGain = [12.5102, 12.5102, 34.4285, 34.4285];
        UnitMWGain = [1,1,1/2.75, 1/2.75];
        %Calculates speed, acceleration, angle
        [NFSPD, PUNFACL, NFANGL, mac_trip_states] = unitCalc(mac_trip_states,
UnitSpeedGain, UnitMWGain,k);
%% Center of Mass Calculations
        %Averages speed, acceleration, and angle
        speed = centerMass(NFSPD(1), NFSPD(2), NFSPD(3), NFSPD(4),
onlineStatus);
        accel = centerMass(PUNFACL(1), PUNFACL(2), PUNFACL(3), PUNFACL(4),
onlineStatus);
        angle = centerMass(NFANGL(1), NFANGL(2), NFANGL(3), NFANGL(4),
onlineStatus);
        %Save values in storage matrix
        mac_trip_states(61,k) = speed;
        mac_trip_states(62,k) = accel;
```

```
        mac_trip_states(63,k) = angle;
%% Look-Ahead Angle Calculation
        lookAheadAng = lookAhead(speed, accel, angle);  %Performs LA Angle
calculation
        mac_trip_states(64,k) = lookAheadAng;           %Save in storage
matrix
%% Trip Logic
        %Trip percentage saved in storage matrix
        mac_trip_states = tripAlgorithms(t(kT), speed, accel, angle,
lookAheadAng, mac_trip_states, k);
%% Trip Arbitration (for 48% trip)
        %Comment out to run in open-loop
        if mac_trip_states(60,k) == 0.48
            tripOut([14, 36]) = true;     %Trip small and large gen
            mac_trip_states([7,17],k) = 1;
            mac_trip_states(65,1) = bin2dec('0101');   %Change online status
        end
    end
end
end
```

14. Appendix E: Unit Calculations Function

```
function [NFSPD, PUNFACL, NFANGL, mac_trip_states] =
unitCalc(mac_trip_states, UnitSpeedGain, UnitMWGain,k)
%Purpose: Perform Unit Calculations
%
%Inputs:
%       mac_trip_states = storage matrix
%       UnitSpeedGain = 1x4 vector containing unit speed gains
%       UnitMWGain = 1x4 vector containing unit power gains
%       k = filter index
%
%Outputs:
%       Speed, acceleration, and angle = 1x4 vectors containing values for
%           each unit
%       mac_trip_states = storage matrix
%
% Version 1.0
% Author:  RJ Hallett
% Date:   March 2017
load filters.mat
SpeedG = 660;
PmechG = 8;
%R - Acceleration from last iteration
R = mac_trip_states(47:50,k-1);
PTIMWoffset = mac_trip_states(4:5:19,1);    %Power Offset
%P1 - Power deviation signal
mac_trip_states([26:29],k) = mac_trip_states([4:5:19],k-1)-PTIMWoffset;
%S1 Speed Deviation*Speed Gain
mac_trip_states([23, 24, 25, 26],k) = (mac_trip_states([6, 11, 16, 21],k-2)-
1)*SpeedG;
for i = 1:4 %For each generator
    %P2 - Filter 1 output
    mac_trip_states = funFiltMacTrip(G1.num{1},G1.den{1},mac_trip_states,
26+i, 30+i,k);
    %P3 - Add back power offset
    P3(i) = mac_trip_states(30+i,k)+PTIMWoffset(i);
    %S2 - Filter 2 output
    mac_trip_states = funFiltMacTrip(G3.num{1},G3.den{1},mac_trip_states,
22+i, 34+i,k);
    NFSPD(i) = mac_trip_states(34+i,k);     %Speed output
    %Filter 3 output
    mac_trip_states = funFiltMacTrip(G5.num{1},G5.den{1},mac_trip_states,
34+i, 38+i,k);
    NFANGL(i) = mac_trip_states(38+i,k);    %Angle output
    %A1 - Filter 4 output
    mac_trip_states = funFiltMacTrip(G4.num{1},G4.den{1},mac_trip_states,
34+i, 42+i,k);
    %A2 - Multiply A1 by speed gains, subtract by previous Accel value,
multiply by mech power gain
    A2(i) = (mac_trip_states(42+i,k)*UnitSpeedGain(i)-R(i))*PmechG;
    %A3 - Add mech accel to electric accel, subtract by power offset
    mac_trip_states(50+i,k) = A2(i) + P3(i) - PTIMWoffset(i);
    %A4 - Filter 5 output
    mac_trip_states = funFiltMacTrip(G2.num{1}, G2.den{1}, mac_trip_states,
50+i,54+i,k);
```

```
    %A5 - Add back offset, subtract by P3
    A5(i) = mac_trip_states(54+i,k)+PTIMWoffset(i) - P3(i);
    mac_trip_states(46+i,k) = A5(i);        %Save A5 for next itereation
    PUNFACL(i) = UnitMWGain(i)*A5(i);       %Multiply by MW gains, set to
Accel output
end
```

15. Appendix F: Center of Mass Function

```
function [cmVal] = centerMass(u1val, u2val, u3val, u4val, onlineStatus)
%Purpose: Calculate the center of mass average
%
%Inputs:
%       uXval = Speed, Accel, or Angle value corresponding to the unit X,
%           where X is a number from 1 - 4
%       onlineStatus = decimal representation of online status
%Outputs:
%       cmVal = center of mass value
%
% Version 1.0
% Author:   RJ Hallett
% Date:   March 2017
%
%Calculate inidvidual unit status:
u1stat = double(bitand(1, onlineStatus));       %bitwise AND to calculate gen
1 status
u2stat = (1/2)*double(bitand(2, onlineStatus)); %bitwise AND to calculate gen
2 status
u3stat = (1/4)*double(bitand(4, onlineStatus)); %bitwise AND to calculate gen
3 status
u4stat = (1/8)*double(bitand(8, onlineStatus)); %bitwise AND to calculate gen
4 status

%Center of mass calculation
cmVal = (u1stat*u1val + u2stat*u2val + 2.75*(u3stat*u3val +
u4stat*u4val))/(u1stat + u2stat + 2.75*(u3stat+u4stat));
end
```

16. Appendix G: Look Ahead Angle Function

```
function [lookAheadAng] = lookAhead(speed, accel, angle)
%Purpose: Calculate look-ahead angle
%
%Inputs:
%       Speed = average speed
%       Accel = average acceleration
%       Angle = average angle
%
%Outputs:
%       lookAheadAng = look-ahead angle
%
% Version 1.0
% Author:   RJ Hallett
% Date:   March 2017
%
%LA angle calculation:
lookAheadAng = speed*15+accel*35.971+angle;
end
```

17. Appendix H: Trip Algorithms Function

```
function [mac_trip_states] = tripAlgorithms(time_in, speed_in,
acceleration_in, angle_in, look_ahead_angle_in, mac_trip_states, k)
%Purpose: Detect events and issue trip percentage
%
%Inputs:
%       time_in = simulation time, angle_in = center of mass angle,
%       look-ahead angle = center of mass look-ahead angle,
%       mac_trip_states = storage matrix, k = filter index
%
%Output:
%       mac_trip_states = storage matrix with trip percentage placed in row
60
% Version 1.0
% Author:   RJ Hallett
% Date:   March 2017
%
%% Gross Enable
[time_out, speed_out, acceleration_out, angle_out, look_ahead_angle_out,
reset]=...
                    gross_enable(time_in, speed_in, acceleration_in,
angle_in, look_ahead_angle_in);
%Rename variables
sim_time = time_out; center_of_mass_speed = speed_out; mac_trip_states(59,k)
= acceleration_out;
center_of_mass_acceleration = acceleration_out; center_of_mass_angle =
angle_out;
center_of_mass_look_ahead_angle = look_ahead_angle_out;
if k>4
    center_of_mass_acceleration_4cycle_delay = mac_trip_states(59, k-4);
%Creates 4 cycle delayed acceleration value
else
    center_of_mass_acceleration_4cycle_delay = acceleration_out;
end
%% Trip Algorithms
%TDAC (Time Triggered Acceleration)
tripPrct1 = tdac(sim_time, center_of_mass_acceleration, center_of_mass_angle,
reset, center_of_mass_acceleration_4cycle_delay);
%OACL (Over-Acceleration)
tripPrct2 = oacl(sim_time, center_of_mass_acceleration, center_of_mass_angle,
reset);
%OSPD (Over-Speed)
tripPrct3 = ospd(sim_time, center_of_mass_speed, reset);
%OANG (Over-Angle)
tripPrct4 = oang(sim_time, center_of_mass_look_ahead_angle, reset);
%ACAN (Acceleration vs LA Angle)
tripPrct5 = acan(sim_time, center_of_mass_acceleration,
center_of_mass_look_ahead_angle, reset);
%SPAN (Speed vs LA Angle)
tripPrct6 = span(sim_time, center_of_mass_speed,
center_of_mass_look_ahead_angle, reset);
%ACSP (Acceleration vs Speed)
tripPrct7 = acsp(sim_time, center_of_mass_acceleration, center_of_mass_speed,
center_of_mass_angle, reset);
%ANAC (Angle Triggered Acceleration)
```

```
tripPrct8 = anac(sim_time, center_of_mass_acceleration, center_of_mass_angle,
reset);
%Second Peak Over-Speed
tripPrct9 = second_peak_over_speed(sim_time, center_of_mass_speed);
%Dynamic Oscillation
tripPrct10 = dynamic_oscillation(sim_time, center_of_mass_speed, reset);
%% Save maximum trip percentage
mac_trip_states(60,k) = max([tripPrct1, tripPrct2, tripPrct3, tripPrct4,
tripPrct5, tripPrct6, tripPrct7, tripPrct8, tripPrct9, tripPrct10]);
end

%%
```

18. Appendix I: Individual Trip Algorithms

Gross Enable:

```
function [time_out, speed_out, acceleration_out,...
                    angle_out, look_ahead_angle_out, reset]=...
                    gross_enable(time_in, speed_in, ...
                    acceleration_in, angle_in, look_ahead_angle_in)

    persistent start_time;
    persistent enable;
    persistent speed_up;
    persistent speed_reset;
    persistent angle_up;
    persistent angle_reset;

    ACCELERATION_LIMIT = 20;
    RESET_LIMIT = 2.5;
    ANG_NOISE_TRSH = 1500.0;
    SPD_NOISE_TRSH = 50.0;

    % Initialize
    if isempty(start_time)
        start_time = 99999;
        enable = 0;
        speed_up = 0;
        speed_reset = 0;
        angle_up = 0;
        angle_reset = 0;
    end

    time_out = 0;
    speed_out = 0;
    acceleration_out = 0;
    angle_out = 0;
    look_ahead_angle_out = 0;
    reset = 0;

    % Disable/Reset
    if(enable)
        % Determine if angle above noise threshold
        if angle_in > ANG_NOISE_TRSH
            angle_up = 1;
            angle_reset = 0;
        end

        % Determine if angle below noise threshold following an up
        if and(angle_in < -ANG_NOISE_TRSH, angle_up)
            angle_reset = 1;
        end

        % Determine if speed above noise threshold
        if speed_in > SPD_NOISE_TRSH
            speed_up = 1;
```

```
            speed_reset = 0;
        end

        % Determine if speed below noise threshold following an up
        if and(speed_in < -SPD_NOISE_TRSH, speed_up)
            speed_reset = 1;
        end

        % Reset if both speed and angle below noise
        if and(speed_reset,angle_reset)
            start_time = 99999;
            enable = 0;
            reset = 1;
            speed_up = 0;
            speed_reset = 0;
            angle_up = 0;
            angle_reset = 0;
            disp('-----------------------------');
            disp('*ATR* Gross Enable - RESET LIMIT HIT - Speed and Angle
below normal noise');
            disp('*ATR* Time:')
            disp(time_in);
        end


        % Reset after reset time limit exceeded
        if (time_in - start_time) > RESET_LIMIT
            start_time = 99999;
            enable = 0;
            reset = 1;
            speed_up = 0;
            speed_reset = 0;
            angle_up = 0;
            angle_reset = 0;
            disp('-----------------------------');
            disp('*ATR* Gross Enable - RESET LIMIT HIT - Timer reset');
            disp('*ATR* Time:')
            disp(time_in);
        end

    end

    % Enable
    if and(acceleration_in > ACCELERATION_LIMIT, not(enable))
        enable = 1;
        start_time = time_in;
        speed_up = 0;
        speed_reset = 0;
        angle_up = 0;
        angle_reset = 0;
        disp('-----------------------------');
        disp('*ATR* Gross Enable - ENABLED');
        disp('*ATR* Time:')
        disp(time_in);
    end
```

```
    if enable
        time_out = time_in;
        speed_out = speed_in;
        acceleration_out = acceleration_in;
        angle_out = angle_in;
        look_ahead_angle_out = look_ahead_angle_in;
    end

end
```

TDAC:

```
function trip_percent = tdac(sim_time, center_of_mass_acceleration,
center_of_mass_angle, reset, center_of_mass_acceleration_4cycle_delay)
    persistent peak_acceleration;
    persistent tdac_start_time;
    persistent all_tdac_complete;
    persistent tdac_10_enable;
    persistent tdac_12_enable;
    persistent tdac_8_enable;
    persistent tdac_7_enable;
    persistent tdac_6_enable;
    persistent tdac_5_enable;
    persistent enable;



    RESET_LIMIT = .5;
    ACCELERATION_LIMIT = 20;
    ANGLE_LIMIT = 4000;
    DELAY_ACCELERATION_LIMIT = 10;


    % Initialize
    if isempty(peak_acceleration)
        peak_acceleration = 0;
        tdac_start_time = 99999;
        all_tdac_complete = 0;
        tdac_12_enable = 1;
        tdac_10_enable = 1;
        tdac_8_enable = 1;
        tdac_7_enable = 1;
        tdac_6_enable = 1;
        tdac_5_enable = 1;
        enable = 0;
%         disp('*ATR* I am empty - init')
    end
    trip_percent = 0;

    % Update maximum center of mass acceleration value
    if center_of_mass_acceleration > peak_acceleration
        peak_acceleration = center_of_mass_acceleration;
    end

    % Reset trip percentage and peak acceleration after reset limit
    if or((sim_time - tdac_start_time) > RESET_LIMIT, reset)
        peak_acceleration = 0;
        tdac_start_time = 99999;
        all_tdac_complete = 0;
        tdac_12_enable = 1;
        tdac_10_enable = 1;
        tdac_8_enable = 1;
        tdac_7_enable = 1;
        tdac_6_enable = 1;
        tdac_5_enable = 1;
```

```
        enable = 0;
 %       disp('*ATR* I am empty - reset')

    end

    % Return if not enabled and acceleration is below limit
    if and(center_of_mass_acceleration < ACCELERATION_LIMIT, not(enable))
        return
    end

    % Return if not enabled and acceleration has not increased over limit in
4 cycles
    if and(((center_of_mass_acceleration -
center_of_mass_acceleration_4cycle_delay) <
DELAY_ACCELERATION_LIMIT),not(enable))
        return
    end

    % Return if all TDAC calculations complete
    if all_tdac_complete
        return
    end

    % Enable
    if not(enable)
        enable = 1;
        tdac_start_time = sim_time;
        disp('----------------------------');
        disp('*ATR* TDAC threshold triggered')
        disp('*ATR* Time:')
        disp(sim_time);
        disp('*ATR* Acceleration:')
        disp(center_of_mass_acceleration);
        disp('*ATR* Peak Acceleration:')
        disp(peak_acceleration);
    end

    % Must be after Enable code
    if center_of_mass_angle < ANGLE_LIMIT
%         disp('*ATR* Angle not over limit:')
%         disp('*ATR* Angle:')
%         disp(center_of_mass_angle);
        return
    end

    tdac_time = sim_time - tdac_start_time;


    % TDAC 12.0
    if (tdac_time * 60.0) >= 12.0
        if not(tdac_12_enable)
            return
        end

        disp('----------------------------');
```

```
    disp('*ATR* Evaluating TDAC 12.0');
    disp('*ATR* Time:')
    disp(sim_time);
    disp('*ATR* Acceleration:')
    disp(center_of_mass_acceleration);
    disp('*ATR* Peak Acceleration:')
    disp(peak_acceleration);

    tdac_12_enable = 0;

    if peak_acceleration < 25.0
        return
    end
    if center_of_mass_acceleration >= (70 - 0.500 * peak_acceleration)
        trip_percent = 0.48;
        disp('*ATR* Issuing a trip for 48.0% from TDAC 12.0');
        return
    end
    if center_of_mass_acceleration >= (55 - 0.583 * peak_acceleration)
        trip_percent = 0.31;
        disp('*ATR* Issuing a trip for 31.0% from TDAC 12.0');
        return
    end
    if center_of_mass_acceleration >= (10)
        trip_percent = 0.31;
        disp('*ATR* Issuing a trip for 31.0% from TDAC 12.0');
        return
    end
end


% TDAC 10.0
if (tdac_time * 60.0) >= 10.0
    if not(tdac_10_enable)
        return
    end

    disp('-----------------------------');
    disp('*ATR* Evaluating TDAC 10.0');
    disp('*ATR* Time:')
    disp(sim_time);
    disp('*ATR* Acceleration:')
    disp(center_of_mass_acceleration);
    disp('*ATR* Peak Acceleration:')
    disp(peak_acceleration);

    tdac_10_enable = 0;

    if peak_acceleration < 25.0
        return
    end
    if center_of_mass_acceleration >= (80 - 0.525 * peak_acceleration)
        trip_percent = 0.48;
        disp('*ATR* Issuing a trip for 48.0% from TDAC 10.0');
        return
```

```
    end
    if center_of_mass_acceleration >= (50 - 0.500 * peak_acceleration)
        trip_percent = 0.31;
        disp('*ATR* Issuing a trip for 31.0% from TDAC 10.0');
        return
    end
    if center_of_mass_acceleration >= (13)
        trip_percent = 0.31;
        disp('*ATR* Issuing a trip for 31.0% from TDAC 10.0');
        return
    end
end


% TDAC 8.0
if (tdac_time * 60.0) >= 8.0
    if not(tdac_8_enable)
        return
    end

    disp('----------------------------');
    disp('*ATR* Evaluating TDAC 8.0');
    disp('*ATR* Time:')
    disp(sim_time);
    disp('*ATR* Acceleration:')
    disp(center_of_mass_acceleration);
    disp('*ATR* Peak Acceleration:')
    disp(peak_acceleration);

    tdac_8_enable = 0;

    if peak_acceleration < 25.0
        return
    end
    if center_of_mass_acceleration >= (80 - 0.300 * peak_acceleration)
        trip_percent = 0.48;
        disp('*ATR* Issuing a trip for 48.0% from TDAC 8.0');
        return
    end
    if center_of_mass_acceleration >= (25 - 0.050 * peak_acceleration)
        trip_percent = 0.31;
        disp('*ATR* Issuing a trip for 31.0% from TDAC 8.0');
        return
    end
end

% TDAC 6.8
if (tdac_time * 60.0) >= 6.8
    if not(tdac_7_enable)
        return
    end

    disp('----------------------------');
    disp('*ATR* Evaluating TDAC 6.8');
    disp('*ATR* Time:')
```

```
    disp(sim_time);
    disp('*ATR* Acceleration:')
    disp(center_of_mass_acceleration);
    disp('*ATR* Peak Acceleration:')
    disp(peak_acceleration);

    tdac_7_enable = 0;

    if peak_acceleration < 75.0
        return
    end
    if center_of_mass_acceleration >= (80 + 0.150 * peak_acceleration)
        trip_percent = 0.81;
        disp('*ATR* Issuing a trip for 81.0% from TDAC 6.8');
        return
    end
    if center_of_mass_acceleration >= (83)
        trip_percent = 0.64;
        disp('*ATR* Issuing a trip for 64.0% from TDAC 6.8');
        return
    end
    if center_of_mass_acceleration >= (53)
        trip_percent = 0.48;
        disp('*ATR* Issuing a trip for 48.0% from TDAC 6.8');
        return
    end
end

% TDAC 6.0
if (tdac_time * 60.0) >= 6.0
    if not(tdac_6_enable)
        return
    end
    disp('----------------------------');
    disp('*ATR* Evaluating TDAC 6.0');
    disp('*ATR* Time:')
    disp(sim_time);
    disp('*ATR* Acceleration:')
    disp(center_of_mass_acceleration);
    disp('*ATR* Peak Acceleration:')
    disp(peak_acceleration);
    tdac_6_enable = 0;

    if peak_acceleration < 75.0
        return
    end
    if center_of_mass_acceleration >= (50 + 0.450 * peak_acceleration)
        trip_percent = 0.81;
        disp('*ATR* Issuing a trip for 81.0% from TDAC 6.0');
        return
    end
    if center_of_mass_acceleration >= (60 + 0.300 * peak_acceleration)
        trip_percent = 0.64;
        disp('*ATR* Issuing a trip for 64.0% from TDAC 6.0');
        return
    end
```

```
    if center_of_mass_acceleration >= (103)
        trip_percent = 0.64;
        disp('*ATR* Issuing a trip for 64.0% from TDAC 6.0');
        return
    end
    if center_of_mass_acceleration >= (40.8 + 0.256 * peak_acceleration)
        trip_percent = 0.48;
        disp('*ATR* Issuing a trip for 48.0% from TDAC 6.0');
        return
    end
    if center_of_mass_acceleration >= (85)
        trip_percent = 0.48;
        disp('*ATR* Issuing a trip for 48.0% from TDAC 6.0');
        return
    end
end


% TDAC 5.2
if (tdac_time * 60.0) >= 5.2
    if not(tdac_5_enable)
        return
    end

    disp('----------------------------');
    disp('*ATR* Evaluating TDAC 5.2');
    disp('*ATR* Time:')
    disp(sim_time);
    disp('*ATR* Acceleration:')
    disp(center_of_mass_acceleration);
    disp('*ATR* Peak Acceleration:')
    disp(peak_acceleration);

    tdac_5_enable = 0;

    if peak_acceleration < 25.0
        return
    end
    if center_of_mass_acceleration >= (115 + 0.225 * peak_acceleration)
        trip_percent = .81;
        disp('*ATR* Issuing a trip for 81.0% from TDAC 5.2');
        return
    end
    if center_of_mass_acceleration >= (25 + 0.714 * peak_acceleration)
        trip_percent = 0.64;
        disp('*ATR* Issuing a trip for 64.0% from TDAC 5.2');
        return
    end
    if center_of_mass_acceleration >= (148)
        trip_percent = 0.64;
        disp('*ATR* Issuing a trip for 64.0% from TDAC 5.2');
        return
    end
end

end
```

OACL:

```
function trip_percent = oacl(sim_time, center_of_mass_acceleration,
center_of_mass_angle, reset)
    persistent trip_issued;

    if isempty(trip_issued)
        trip_issued = 0;
    end

    if reset
        trip_issued = 0;
    end

    trip_percent = 0;

    % Return if trip already issued
    if trip_issued
        return
    end

    if and(center_of_mass_acceleration >= 190, center_of_mass_angle >= 4000)
        trip_percent = 0.81;
        trip_issued = 1;

        disp('-----------------------------');
        disp('*ATR* Evaluating OACL');
        disp('*ATR* Time:')
        disp(sim_time);
        disp('*ATR* Acceleration:')
        disp(center_of_mass_acceleration);
        disp('*ATR* Angle:')
        disp(center_of_mass_angle);
        disp('*ATR* Issuing a trip for 81.0% from OACL');
    end

end
```

OSPD:

```
function trip_percent = ospd(sim_time, center_of_mass_speed, reset)
    persistent trip_issued;

    if isempty(trip_issued)
        trip_issued = 0;
    end

    if reset
        trip_issued = 0;
    end

    trip_percent = 0;

    % Return if trip already issued
    if trip_issued
        return
    end

    if center_of_mass_speed >= 840
        trip_percent = 0.48;
        trip_issued = 1;

        disp('-----------------------------');
        disp('*ATR* Evaluating OSPD');
        disp('*ATR* Time:')
        disp(sim_time);
        disp('*ATR* Speed:')
        disp(center_of_mass_speed);
        disp('*ATR* Issuing a trip for 48.0% from OSPD');
    end

end
```

OANG:

```
function trip_percent = ospd(sim_time, center_of_mass_speed, reset)
    persistent trip_issued;

    if isempty(trip_issued)
        trip_issued = 0;
    end

    if reset
        trip_issued = 0;
    end

    trip_percent = 0;

    % Return if trip already issued
    if trip_issued
        return
    end

    if center_of_mass_speed >= 840
        trip_percent = 0.48;
        trip_issued = 1;

        disp('-----------------------------');
        disp('*ATR* Evaluating OSPD');
        disp('*ATR* Time:')
        disp(sim_time);
        disp('*ATR* Speed:')
        disp(center_of_mass_speed);
        disp('*ATR* Issuing a trip for 48.0% from OSPD');
    end

end
```

ACAN:

```
function trip_percent = acan(sim_time, center_of_mass_acceleration, ...
                             center_of_mass_look_ahead_angle, reset)
    persistent trip_issued;
    persistent peak_acceleration;

    if isempty(trip_issued)
        trip_issued = 0;
        peak_acceleration = 0;
    end

    if reset
        trip_issued = 0;
        peak_acceleration = 0;
    end

    trip_percent = 0;

    % Update maximum center of mass acceleration value
    if center_of_mass_acceleration > peak_acceleration
        peak_acceleration = center_of_mass_acceleration;
    end

    % Return if trip already issued
    if trip_issued
        return
    end

    if center_of_mass_look_ahead_angle >= (12000 + 162.5 * peak_acceleration)
        trip_percent = 0.14;
        trip_issued = 1;

        disp('----------------------------');
        disp('*ATR* Evaluating ACAN');
        disp('*ATR* Time:')
        disp(sim_time);
        disp('*ATR* Look Ahead Angle:')
        disp(center_of_mass_look_ahead_angle);
        disp('*ATR* Peak Acceleration:')
        disp(peak_acceleration);
        disp('*ATR* Issuing a trip for 14.0% from ACAN');
    end

end
```

SPAN:

```
function trip_percent = span(sim_time, center_of_mass_speed, ...
                             center_of_mass_look_ahead_angle, reset)
    persistent trip_issued;
    persistent peak_speed;

    if isempty(trip_issued)
        trip_issued = 0;
        peak_speed = 0;
    end

    if reset
        trip_issued = 0;
        peak_speed = 0;
    end

    trip_percent = 0;

    % Update maximum center of mass speed value
    if center_of_mass_speed > peak_speed
        peak_speed = center_of_mass_speed;
    end

    % Return if trip already issued
    if trip_issued
        return
    end

    if center_of_mass_look_ahead_angle >= (12000 + 32.5 * peak_speed)
        trip_percent = 0.14;
        trip_issued = 1;

        disp('----------------------------');
        disp('*ATR* Evaluating SPAN');
        disp('*ATR* Time:')
        disp(sim_time);
        disp('*ATR* Look Ahead Angle:')
        disp(center_of_mass_look_ahead_angle);
        disp('*ATR* Peak Speed:')
        disp(peak_speed);
        disp('*ATR* Issuing a trip for 14.0% from SPAN');
    end

end
```

ACSP:

```
function trip_percent = acsp(sim_time, center_of_mass_acceleration, ...
                             center_of_mass_speed, center_of_mass_angle,
reset)
    persistent trip_issued;
    persistent peak_acceleration;

    ANGLE_LIMIT = 4000;

    if isempty(trip_issued)
        trip_issued = 0;
        peak_acceleration = 0;
    end

    if reset
        trip_issued = 0;
        peak_acceleration = 0;
    end

    trip_percent = 0;

    % Update maximum center of mass acceleration value
    if center_of_mass_acceleration > peak_acceleration
        peak_acceleration = center_of_mass_acceleration;
    end

    % Return if trip already issued
    if trip_issued
        return
    end

    % Must be after Enable code
    if center_of_mass_angle < ANGLE_LIMIT
        return
    end

    if center_of_mass_speed >= (170 + 3.6 * peak_acceleration)
        trip_percent = 0.48;
        trip_issued = 1;

        disp('-----------------------------');
        disp('*ATR* Evaluating ACSP');
        disp('*ATR* Time:')
        disp(sim_time);
        disp('*ATR* Speed:')
        disp(center_of_mass_speed);
        disp('*ATR* Peak Acceleration:')
        disp(peak_acceleration);
        disp('*ATR* Issuing a trip for 48.0% from ACSP');
    end

end
```

ANAC:

```
function trip_percent = anac(sim_time, center_of_mass_acceleration, ...
                             center_of_mass_angle, reset)
    persistent trip_issued;
    persistent peak_acceleration;

    if isempty(trip_issued)
        trip_issued = 0;
        peak_acceleration = 0;
    end

    if reset
        trip_issued = 0;
        peak_acceleration = 0;
    end

    trip_percent = 0;

    % Update maximum center of mass acceleration value
    if center_of_mass_acceleration > peak_acceleration
        peak_acceleration = center_of_mass_acceleration;
    end

    % Return if trip already issued
    if trip_issued
        return
    end

    if center_of_mass_angle >= 16000
        trip_issued = 1;

        disp('-----------------------------');
        disp('*ATR* Evaluating ANAC');
        disp('*ATR* Time:')
        disp(sim_time);
        disp('*ATR* Acceleration:')
        disp(center_of_mass_acceleration);
        disp('*ATR* Peak Acceleration:')
        disp(peak_acceleration);
        disp('*ATR* Angle:')
        disp(center_of_mass_angle);

        if center_of_mass_acceleration >= (215 - 0.750 * peak_acceleration)
            trip_percent = 1.00;
            disp('*ATR* Issuing a trip for 100.0% from ANAC');
            return
        end
        if center_of_mass_acceleration >= (46)
            trip_percent = 0.81;
            disp('*ATR* Issuing a trip for 81.0% from ANAC');
            return
        end
        if center_of_mass_acceleration >= (75.33 - 0.3167 *
peak_acceleration)
```

```
            trip_percent = 0.64;
            disp('*ATR* Issuing a trip for 64.0% from ANAC');
            return
        end
        if center_of_mass_acceleration >= (-4)
            trip_percent = 0.48;
            disp('*ATR* Issuing a trip for 48.0% from ANAC');
            return
        end
        if center_of_mass_acceleration >= (20 - 0.30 * peak_acceleration)
            trip_percent = 0.48;
            disp('*ATR* Issuing a trip for 48.0% from ANAC');
            return
        end
        if center_of_mass_acceleration >= (32 - 0.45 * peak_acceleration)
            trip_percent = 0.31;
            disp('*ATR* Issuing a trip for 31.0% from ANAC');
            return
        end
    end

end
```

Second Peak Over Speed:

```
function trip_percent = second_peak_over_speed(sim_time, ...
                            center_of_mass_speed)
    persistent trip_issued;
    persistent peak_speed;
    persistent first_peak;
    persistent reset;

    if isempty(trip_issued)
        trip_issued = 0;
        peak_speed = 0;
        first_peak = 0;
        i = 0;
        reset = 0;
    end

    if reset
        trip_issued = 0;
        peak_speed = 0;
        first_peak = 0;
        i = 0;
    end

    trip_percent = 0;

    % Update maximum center of mass speed value
    if center_of_mass_speed > peak_speed
        peak_speed = center_of_mass_speed;
    end

    % Return if trip already issued
    if trip_issued
        return
    end

    if center_of_mass_speed < 0;
        reset = 0;
        return
    end


    if and(not(first_peak),center_of_mass_speed < peak_speed)
        test_value = 12;
        if (0.8 * peak_speed) > test_value
            test_value = (0.8 * peak_speed);
        end

        if center_of_mass_speed < test_value
            disp('----------------------------');
            disp('*ATR* SECOND-PEAK-OVER-SPEED - Peak Detected');
            disp('*ATR* Time:')
            disp(sim_time);
            disp('*ATR* Peak Speed:')
```

```
            disp(peak_speed);
            first_peak = peak_speed;
            peak_speed = 0;
        end
        return
    end

    if and(first_peak,center_of_mass_speed > first_peak)
        disp('----------------------------');
        disp('*ATR* Evaluating SECOND-PEAK-OVER-SPEED');
        disp('*ATR* Time:')
        disp(sim_time);
        disp('*ATR* First Peak:')
        disp(first_peak);
        disp('*ATR* Current Speed:')
        disp(center_of_mass_speed);
        if first_peak >= 450
            trip_issued = 1;
            trip_percent = 0.81;
            disp('*ATR* Issuing a trip for 81.0% from SECOND-PEAK-OVER-
SPEED');
            return
        end
        if first_peak >= 175
            trip_issued = 1;
            trip_percent = 0.48;
            disp('*ATR* Issuing a trip for 48.0% from SECOND-PEAK-OVER-
SPEED');
            return
        end
        first_peak = 0;

    end
end
```

Dynamic Oscillation:

```
function trip_percent = dynamic_oscillation(sim_time, ...
                            center_of_mass_speed, reset)
    persistent trip_issued;
    persistent peak_count;
    persistent min_peak;
    persistent max_peak;
    persistent max_amplitude;
    persistent first_peak;
    persistent second_peak_enable;
    persistent previous_speed;

    if isempty(trip_issued)
        trip_issued = 0;
        min_peak = 0;
        max_peak = 0;
        max_amplitude = 0;
        first_peak = 0;
        second_peak_enable = 0;
        peak_count = 0;
    end

    if reset
        trip_issued = 0;
        min_peak = 0;
        max_peak = 0;
        max_amplitude = 0;
        first_peak = 0;
        second_peak_enable = 0;
        peak_count = 0;
    end

    trip_percent = 0;

    % Update maximum center of mass speed value
    if center_of_mass_speed > max_peak
        max_peak = center_of_mass_speed;
    end

    % Update maximum center of mass speed value
    if center_of_mass_speed < min_peak
        min_peak = center_of_mass_speed;
    end

    delta = max_peak - min_peak;
    if delta > max_amplitude
        max_amplitude = delta;
    end

    % Return if trip already issued
    if trip_issued
        return
    end
```

```
    if and(previous_speed == min_peak, center_of_mass_speed > previous_speed)
        second_peak_enable = 1;
        peak_count = peak_count + 1;
    end


    if second_peak_enable
        if max_amplitude > first_peak
            if first_peak >= 450
                trip_issued = 1;
                trip_percent = 0.81;
                disp(sim_time);
                disp('*ATR* Issuing a trip for 81.0% from SECOND-PEAK-OVER-
SPEED');
                return
            end
            if first_peak >= 175
                trip_issued = 1;
                trip_percent = 0.48;
                disp(sim_time);
                disp('*ATR* Issuing a trip for 48.0% from SECOND-PEAK-OVER-
SPEED');
                return
            end
        end
    else
        if max_amplitude > 0
           first_peak = max_amplitude;
        else
            if center_of_mass_speed < 0
                second_peak_enable = 1;
                peak_speed = 0;
            end
        end
    end

    previous_speed = center_of_mass_speed;
end
```

19. Appendix J: Run Simulations Code for ATR

```
clear all; close all; clc
PSTpath = 'F:\Research\ATR\PSTcode\pstV2p2';%'..\pstV2';
addpath(PSTpath)
save delme PSTpath

%% Scenario 1
clear all; close all; clc
load delme
delete([PSTpath '\DataFile.m']);
copyfile('d_minniwecc_v3c_c3_6_c_ATR_local_fault_thesis.m',[PSTpath
'\DataFile.m']);
delete([PSTpath '\mac_trip_logic.m']);
copyfile('mac_trip_logic_ATRV3_thesis.m',[PSTpath '\mac_trip_logic.m']);
delete([PSTpath '\mac_sub.m']);
copyfile([PSTpath '\mac_sub_NEW2.m'],[PSTpath '\mac_sub.m']);
delete([PSTpath '\ml_sig.m']);
copyfile('ml_sig_bus20.m',[PSTpath '\ml_sig.m']);
s_simu_Batch
% save d_microWECC_V3C_C3_6_C_AlbertaSw_V9_6_1_1_BRAKE_2
Altpath = 'F:\Research\ATR\ATR';
%save([Altpath
'\d_minniwecc_v3c_c3_6_c_ATR_10cycle_fault_openloop_thesis_kfix.mat'])
```

20. Appendix K: Run Simulink ATR Code

```
% Run PST ATR Model in Simulink
clear
load 'd_minniwecc_v3c_c3_6_c_ATR_10cycle_fault_openloop_thesis_kfix.mat';

Timesec = 0:1/60:5-1/60;

U1PTIMW = mac_trip_states(4,1:length(Timesec))';
U2PTIMW = mac_trip_states(9,1:length(Timesec))';
U3PTIMW = mac_trip_states(14,1:length(Timesec))';
U4PTIMW = mac_trip_states(19,1:length(Timesec))';

U1PTIVEL = mac_trip_states(6,1:length(Timesec))';
U2PTIVEL = mac_trip_states(11,1:length(Timesec))';
U3PTIVEL = mac_trip_states(16,1:length(Timesec))';
U4PTIVEL = mac_trip_states(21,1:length(Timesec))';

U1PTIVEL = U1PTIVEL-1;
U2PTIVEL = U2PTIVEL-1;
U3PTIVEL = U3PTIVEL-1;
U4PTIVEL = U4PTIVEL-1;


for k = 1:length(Timesec)
ONLINESTAT(k) = 15;
end
ONLINESTAT = ONLINESTAT';
Timesec = Timesec';
sim final_main_clean.slx
```

21. Appendix L: MiniWECC PST Data File

```
% d_minniWECC_V3C_C3_6.m
% PST data file
%
% MinniWECC system, version 3C, case 3, number 6.  Same as
d_minniWECC_V3C_C3_5.m but
% added PSS units as a variable and Alberta connection as a variable.
Alberta is disconnected by removing gen 34,
% removing the load on bus 120, and reducing the line impedances to 1% (this
% allows me to maintain the same bus numbers).
%
% MT_P reduces the flow by reducing the power for Colstrip.  Alberta import
is decreased
% by decreasing the power for Alberta generation.  For both cases, the power
is
% equally increased at gen 7 thru 11.  COI loading is reduced decreasing bus
43 Pload by COI_P,
% gen 7 is reduced as it is the swing bus.
%
% System bases:  100 MVA, 60 Hz.

% Written by Dan Trudnowski, 2011
%function
[bus,line,mac_con,exc_con,pss_con,tg_con,load_con,lmod_con,rlmod_con,lmon_con
,sw_con] = d_minniWECC_V3C_C3_6_fun(COI_P,MT_P,Alberta_P,pssGen,AlbCon)
%Case C
    pssGen = [1;13;14]; %Same as d_minniWECC_V3C_C3_5
    AlbCon = 1; %Alberta connected
    COI_P = 20; MT_P = 5; Alberta_P = 7.5;
%   COI_P = amount to reduce COI by
%   MT_P = amount to reduce Colstrip by
%   Alberta_P = amount to reduce Alberta import by
%   pssGen = vector of generators to equip with pss units
%   AlbCon = If == 1, Alberta is connected; If == 0, Alberta disconnected

%% Operating parameters
% PDC = DC line real-power flows
PDC = [28.5; %PDCI
       19]; %

F = (MT_P + Alberta_P)/5; %Amount to add to gen 7 thru 11

if ~(AlbCon==0 || AlbCon==1)
    error('AlbCon must be 0 or 1');
end

% TCSC (not used)
YTCSCf = 1000; %TCSC fixed series capacitive admittance
YTCSCcMax = 0.2*YTCSCf; %TCSC controllable admittance limit
TCSC_Gen = [10;21]; %Generators' to use for speed feedback
TCSC_Gain = (0/0.1)*60; %TCSC gain in admittance(pu)/speed(pu)

%% Bus data
```

```
% bus data format
% bus:
% col1 number
% col2 voltage magnitude(pu)
% col3 voltage angle(degree)
% col4 p_gen(pu)
% col5 q_gen(pu),
% col6 p_load(pu)
% col7 q_load(pu)
% col8 G shunt(pu)
% col9 B shunt(pu)
% col10 bus_type
%     bus_type  - 1, swing bus
%               - 2, generator bus (PV bus)
%               - 3, load bus (PQ bus)
% col11 q_gen_max(pu)
% col12 q_gen_min(pu)
% col13 v_rated (kV)
% col14 v_max  pu
% col15 v_min  pu
bus = [...
% bus    V        Angle     P_gen     Q_gen     P_load    Q_load          G
B      type QgenMax   QgenMin      V_rate vmax vmin
    1    1.0250   26.7940   22.0000-2    0.3847         0         0         0
0    2    45        -45           20     1.5  0.5; %Gen 1 bus
    2    1.0256   23.5949         0         0   -0.0000    0.0000         0
2.0000    3    0.0        0.0         500    1.5  0.5; %HV bus for Gen 1, BC
    3    1.0250   14.9547   19.2500-2    0.3251         0         0         0
0    2    37        -37           20     1.5  0.5; %Gen 2 bus
    4    1.0258   11.5505         0         0    0.0000   -0.0000         0
2.0000    3    0.0        0.0         500    1.5  0.5; %HV bus for Gen 2, BC
    5    1.0250   31.3527   29.7500-2    3.9890         0         0         0
0    2    37        -37           20     1.5  0.5; %Gen 3 bus
    6    1.0167   26.0405         0         0    0.0000   -0.0000         0
2.0000    3    0.0        0.0         500    1.5  0.5; %HV bus for Gen 3, BC
    7    1.0214   -4.9443         0         0   -0.0000   -0.0000         0
28.0000    3    0.0        0.0         500    1.5  0.5; %HV bus for bus 8
load, BC
    8    1.0063   -8.0116         0         0   44.0000   11.0000         0
0    3    0.0        0.0         110    1.5  0.5; %Load bus
    9    1.0250   -9.2886   12.7500-2    1.4852         0         0         0
0    2    20        -20           20     1.5  0.5; %Gen 4 bus
   10    1.0190  -13.4887         0         0   -0.0000   -0.0000         0
19.5000    3    0.0        0.0         500    1.5  0.5; %HV bus for Gen 4 and
bus 11 load, North
   11    0.9923  -18.5949         0         0   54.0000   13.5000         0
0    3    0.0        0.0         110    1.5  0.5; %Load bus
   12    1.0250    0.0584   27.2000-10    4.6084         0         0
0         0    2    40        -40           20     1.5  0.5; %Gen 5 bus
   13    1.0146   -4.4417         0         0    0.0000    0.0000         0
0    3    0.0        0.0         500    1.5  0.5; %HV bus for Gen 5, North
   14    1.0250  -18.2087    8.5000    1.7107         0         0         0
0    2    12        -12           20     1.5  0.5; %Gen 6 bus
   15    1.0117  -22.9103         0         0    0.0000    0.0000         0
14.0000-3    3    0.0        0.0         500    1.5  0.5; %HV bus for Gen 6
and bus 16 load, North
```

```
   16    0.9847  -28.0933         0         0   36.0000    9.0000         0
0    3    0.0        0.0         110    1.5  0.5; %Load bus
   17    1.0250         0   33.4224+F    3.1060         0         0         0
0    1    83        -83           20     1.5  0.5; %Gen 7 bus (swing bus)
   18    1.0217   -2.6446         0         0   -0.0000   -0.0000         0
0    3    0.0        0.0         500    1.5  0.5; %HV bus for Gen 7, North
   19    1.0250   -6.3650   25.5000+F    2.1979         0         0         0
0    2    36        -36           20     1.5  0.5; %Gen 8 bus
   20    1.0212  -11.0228         0         0    0.0000    0.0000         0
9.2500    3    0.0        0.0         500    1.5  0.5; %HV bus for Gen 8, 9,
and bus 21 load, North
   21    0.9946  -16.1064         0         0    9.0000    2.2500         0
0    3    0.0        0.0         110    1.5  0.5; %Load bus
   22    1.0250   -5.7955    9.7750+F    0.8428         0         0         0
0    2    12.3      -12.3          20     1.5  0.5; %Gen 9 bus
   23    1.0250  -11.6567   32.3000+F    2.3076         0         0         0
0    2    50        -30           20     1.5  0.5; %Gen 10 bus
   24    1.0224  -15.8989         0         0     PDC(1)   0.0000         0
11.1250-3    3    0.0        0.0         500    1.5  0.5; %HV bus for Gen 10,
11, and bus 26 load, North
   25    1.0250  -11.2465   21.2500+F    1.5300         0         0         0
0    2    30        -30           20     1.5  0.5; %Gen 11 bus
   26    0.9958  -20.9704         0         0    4.5000    1.1250         0
0    3    0.0        0.0         110    1.5  0.5; %Load bus
   27    1.0250    7.3655    2.1250    0.5795         0         0         0
0    2     3         -3           20     1.5  0.5; %Gen 12 bus
   28    1.0058    2.6363         0         0    0.0000    0.0000         0
6.9375    3    0.0        0.0         500    1.5  0.5; %HV bus for Gen 12 and
bus 29 load, North
   29    0.9786   -2.6100         0         0   15.7500    3.9375         0
0    3    0.0        0.0         110    1.5  0.5; %Load bus
   30    1.0250   26.5402   12.7500    0.6454         0         0         0
0    2    15        -15           20     1.5  0.5; %Gen 13 bus
   31    1.0242   21.8959         0         0   -0.0000    0.0000         0
0    3    0.0        0.0         500    1.5  0.5; %HV bus for Gen 13, North
   32    1.0250   48.6408   19.2950-MT_P    0.7683         0         0
0         0    2    23          -23           20     1.5  0.5; %Gen 14 bus
   33    1.0258   43.1467         0         0    0.0000   -0.0000         0
3.0000-3    3    0.0        0.0         500    1.5  0.5; %HV bus for Gen 14,
North
   34    1.0250  -26.9292    4.6750    0.3233         0         0         0
0    2    6.5       -6.5          20     1.5  0.5; %Gen 15 bus
   35    1.0226  -31.6521         0         0   -0.0000   -0.0000         0
6.2500    3    0.0        0.0         500    1.5  0.5; %HV bus for Gen 15 and
bus 36 load, North
   36    0.9961  -36.7211         0         0    9.0000    2.2500         0
0    3    0.0        0.0         110    1.5  0.5; %Load bus
   37    1.0250  -50.0781   25.0000    2.1043         0         0         0
0    2    30        -30           20     1.5  0.5; %Gen 16 bus
   38    1.0280  -50.8169         0         0   -0.0000   -0.0000         0
4.0000-4    3    0.0        0.0         500    1.5  0.5; %HV bus for Gen 16,
nWest
   39    1.0250  -63.4051   20.0000    2.2455         0         0         0
0    2    25        -25           20     1.5  0.5; %Gen 17 bus
   40    1.0335  -57.2756         0         0   -0.0000   -0.0000         0
12.0000-6    3    0.0        0.0         500    1.5  0.5; %HV bus, nWest
```

```
   41    1.0250  -64.6948   50.0000    4.5989         0         0         0
0    2    52        -52           20     1.5  0.5; %Gen 18 bus
   42    1.0214  -67.8538         0         0    0.0000    0.0000         0
7.0000-7    3    0.0        0.0         500    1.5  0.5; %HV bus for Gen 18,
nWest
   43    0.9989  -72.6241         0         0  162.4000-COI_P-10   40.6000
0         0    3    0.0        0.0         110    1.5  0.5; %Load bus
   44    1.0188  -68.6798         0         0   -0.0000   -0.0000         0
60.6000-22    3    0.0        0.0         500    1.5  0.5; %HV bus for bus 43
load and gen 17, nWest
   45    1.0250  -52.8607   26.1000    0.7803         0         0         0
0    2    35        -35           20     1.5  0.5; %Gen 19 bus
   46    1.0261  -57.7414         0         0    0.0000    0.0000         0
0    3    0.0        0.0         500    1.5  0.5; %HV bus for Gen 19, sWest
   47    1.0250  -45.8809   19.1400    1.5863         0         0         0
0    2    26        -26           20     1.5  0.5; %Gen 20 bus
   48    1.0250  -57.6897  110.4900    0.3951         0         0         0
0    2    178       -178          20     1.5  0.5; %Gen 21 bus
   49    1.0273  -61.7461         0         0    -PDC(1)   0.0000         0
52.6250    3    0.0        0.0         500    1.5  0.5; %HV bus for Gen 21
and load bus 50, sWest
   50    1.0026  -66.4798         0         0  110.5000   27.6250         0
0    3    0.0        0.0         110    1.5  0.5; %Load bus
   51    1.0250  -70.3719   10.4400    0.4226         0         0         0
0    2    25        -25           20     1.5  0.5; %Gen 22 bus
   52    1.0242  -73.1079         0         0    0.0000   -0.0000         0
2.0000    3    0.0        0.0         230    1.5  0.5; %HV bus for Gen 22,
sWest
   53    1.0250  -84.2165    4.3500    1.4722         0         0         0
0    2    35        -35           20     1.5  0.5; %Gen 23 bus
   54    1.0202  -85.0337         0         0   -0.0000    0.0000         0
18.7125    3    0.0        0.0         500    1.5  0.5; %HV bus for Gen 23
and load bus 55, sWest
   55    0.9952  -89.8359         0         0   34.8500    8.7125         0
0    3    0.0        0.0         110    1.5  0.5; %Load bus
   56    0.9986  -69.6270         0         0   68.0000   17.0000         0
0    3    0.0        0.0         110    1.5  0.5; %Load bus
   57    1.0234  -64.8563         0         0    0.0000    0.0000         0
35.0000    3    0.0        0.0         500    1.5  0.5; %HV bus for Gen 24,
25, load bus 56, sWest
   58    1.0250  -59.5027   26.1000    1.6470         0         0         0
0    2    32        -32           20     1.5  0.5; %Gen 24 bus
   59    1.0250  -59.9156   39.1500    2.3825         0         0         0
0    2    52        -52           20     1.5  0.5; %Gen 25 bus
   60    1.0250  -50.2581   42.3000    4.3692         0         0         0
0    2    47        -47           20     1.5  0.5; %Gen 26 bus
   61    1.0196  -56.1898         0         0   -0.0000   -0.0000         0
5.0000    3    0.0        0.0         500    1.5  0.5; %HV bus for Gen 26,
East
   62    1.0250  -56.2978  100.8000   11.4939         0         0         0
0    2    113       -113          20     1.5  0.5; %Gen 27 bus
   63    1.0185  -62.1833         0         0   -0.0000   -0.0000         0
40.1500    3    0.0        0.0         500    1.5  0.5; %HV bus for Gen 27
and load bus 64, East
   64    0.9917  -67.2955         0         0  120.6000   30.1500         0
0    3    0.0        0.0         110    1.5  0.5; %Load bus
```

```
   65    1.0250  -49.2082   67.5000   -3.3380         0         0         0
0    2    100       -100          20     1.5  0.5; %Gen 28 bus
   66    1.0319  -53.6001         0         0   -0.0000   -0.0000         0
10.0000-10    3    0.0        0.0         500    1.5  0.5; %HV bus for Gen
28, East
   67    1.0249  -65.6940         0         0    0.0000    0.0000         0
14.7500    3    0.0        0.0          500    1.5  0.5; %HV bus for load bus
95, East
   68    1.0250  -42.2342   81.0000    5.5317         0         0         0
0    2    105       -105          20     1.5  0.5; %Gen 29 bus
   69    1.0228  -47.2999         0         0    PDC(2)    0.0000         0
22.6250    3    0.0        0.0          345    1.5  0.5; %HV bus for Gen 29
and load bus 70, East
   70    0.9962  -52.3671         0         0   58.5000   14.6250         0
0    3    0.0        0.0         110    1.5  0.5; %Load bus
   71    1.0250  -44.5135   85.5000   15.5046         0         0         0
0    2    97         -97           20     1.5  0.5; %Gen 30 bus
   72    1.0116  -50.3686         0         0   -0.0000    0.0000         0
20.2500    3    0.0        0.0          345    1.5  0.5; %HV bus for Gen 30
and load bus 73, East
   73    0.9846  -55.5532         0         0   81.0000   20.2500         0
0    3    0.0        0.0         110    1.5  0.5; %Load bus
   74    1.0250  -18.8574   18.7000    0.1526         0         0         0
0    2    24         -24           20     1.1  0.9; %Gen 31 bus
   75    1.0283  -23.9467         0         0    0.0000    0.0000         0
2.0000-2    3    0.0        0.0          345    1.5  0.5; %HV bus for Gen 31,
North
   76    1.0250  -47.5326   15.3000    3.2617         0         0         0
0    2    22         -22           20     1.5  0.5; %Gen 32 bus
   77    1.0109  -52.1521         0         0    0.0000    0.0000         0
11.7500    3    0.0        0.0          230    1.5  0.5; %HV bus for Gen 32
and load bus 78, North
   78    0.9839  -57.3435         0         0   27.0000    6.7500         0
0    3    0.0        0.0         110    1.5  0.5; %Load bus
   79    1.0178    7.0969         0         0   -0.0000   -0.0000         0
3.0000    3    0.0        0.0          500    1.5  0.5; %HV node, BC
   80    1.0118    7.8241         0         0   -0.0000    0.0000         0
2.0000    3    0.0        0.0          500    1.5  0.5; %HV node, BC
   81    1.0163    1.3678         0         0    0.0000   -0.0000         0
1.0000    3    0.0        0.0          500    1.5  0.5;
   82    1.0281  -11.1681         0         0         0         0         0
7.0000    3    0.0        0.0          500    1.5  0.5;
   83    1.0328   19.8541         0         0   -0.0000   -0.0000         0
7.0000    3    0.0        0.0          500    1.5  0.5;
   84    1.0318   13.2766         0         0   -0.0000   -0.0000         0
7.0000    3    0.0        0.0          500    1.5  0.5;
   85    1.0179   24.4253         0         0   -0.0000    0.0000         0
0    3    0.0        0.0         500    1.5  0.5;
   86    1.0229  -32.5634         0         0    0.0000    0.0000         0
2.0000    3    0.0        0.0          500    1.5  0.5;
   87    1.0216  -37.1888         0         0   -0.0000   -0.0000         0
3.0000    3    0.0        0.0          500    1.5  0.5;
   88    1.0154  -35.8480         0         0   -0.0000   -0.0000         0
3.0000    3    0.0        0.0          500    1.5  0.5;
   89    1.0140  -41.6678         0         0    0.0000    0.0000         0
5.0000    3    0.0        0.0          500    1.5  0.5;
```

```
   90    1.0174  -54.8314         0         0   -0.0000    0.0000         0
1.0000    3    0.0        0.0         500    1.5  0.5;
   91    1.0261  -59.7449         0         0   -0.0000    0.0000         0
0    3    0.0        0.0         500    1.5  0.5;
   92    1.0065  -66.1848         0         0   -0.0000    0.0000         0
0    3    0.0        0.0         500    1.5  0.5;
   93    1.0207  -63.5230         0         0   -0.0000   -0.0000         0
0    3    0.0        0.0         500    1.5  0.5;
   94    1.0048  -62.2037         0         0   -0.0000    0.0000         0
0    3    0.0        0.0         345    1.5  0.5; %HV node, x-former, sWest
   95    0.9984  -70.7403         0         0   27.0000    6.7500         0
0    3    0.0        0.0         110    1.5  0.5; %Load bus to HV bus 67
   96    1.0116  -62.9697         0         0   -0.0000    0.0000         0
2.0000    3    0.0        0.0         345    1.5  0.5; %HV node, x-former,
East
   97    1.0223  -57.8634         0         0    0.0000   -0.0000         0
0    3    0.0        0.0         500    1.5  0.5;
   98    1.0227  -59.2048         0         0   -0.0000   -0.0000         0
0    3    0.0        0.0         500    1.5  0.5;
   99    1.0218  -58.5050         0         0    0.0000    0.0000         0
0    3    0.0        0.0         500    1.5  0.5;
  100    1.0146  -43.4281         0         0   -0.0000   -0.0000         0
0    3    0.0        0.0         345    1.5  0.5; %HV node, North-East
  101    1.0111  -39.5113         0         0   -0.0000    0.0000         0
2.0000    3    0.0        0.0         345    1.5  0.5; %HV node, North
  102    1.0080  -40.9162         0         0   -0.0000   -0.0000         0
3.0000    3    0.0        0.0         230    1.5  0.5; %HV node, x-former,
North
  103    1.0203   15.6828         0         0    0.0000   -0.0000         0
1.0000    3    0.0        0.0         230    1.5  0.5; %HV node, x-former,
North
  104    1.0092  -40.5008         0         0   -0.0000   -0.0000         0
0    3    0.0        0.0         345    1.5  0.5; %HV node, x-former, North
  105    1.0148  -39.1549         0         0         0         0         0
1.0000    3    0.0        0.0         500    1.5  0.5; %HV node, x-former,
North
  106    1.0123   22.4614         0         0    0.0000    0.0000         0
0    3    0.0        0.0         500    1.5  0.5; %HV node, Alberta
  107    1.0250  -41.9516   10.0000    1.1583         0         0         0
0    2    12        -12           20     1.5  0.5; %Gen 33 bus
  108    1.0184  -47.4489         0         0   -0.0000    0.0000         0
5.5000    3    0.0        0.0         345    1.5  0.5; %HV bus for Gen 33 and
load bus 109, nWest
  109    0.9984  -51.3965         0         0   14.0000    3.5000         0
0    3    0.0        0.0         110    1.5  0.5; %Load bus
  110    1.0355  -42.4121         0         0   -0.0000    0.0000         0
2.0000    3    0.0        0.0         345    1.5  0.5; %HV node, x-former,
nWest
  111    1.0134  -12.5639         0         0   -0.0000    0.0000         0
0    3    0.0        0.0         230    1.5  0.5; %HV node, x-former, North
  112    0.9996  -80.8948         0         0  131.7500   32.9375         0
0    3    0.0        0.0         110    1.5  0.5; %Load bus
  113    1.0244  -76.1328         0         0   -PDC(2)    0.0000         0
60.9375    3    0.0        0.0         500    1.5  0.5; %HV bus for load bus
112, sWest
  114    1.0205  -63.2552         0         0   -0.0000   -0.0000         0
2.0000    3    0.0        0.0         500    1.5  0.5; %HV node, nWest
```

```
  115    1.0215  -50.7207         0         0    0.0000    0.0000         0
0    3    0.0        0.0         500    1.5  0.5; %HV bus for Gen 20, sWest
  116    1.0189  -70.4970         0         0    0.0000    0.0000         0
0    3    0.0        0.0         500    1.5  0.5; %HV node, sWest
  117    1.0125   22.1819         0         0   -0.0000    0.0000         0
0    3    0.0        0.0         500    1.5  0.5; %HV node, Alberta
  118    1.0250   26.5952   [92.0000-Alberta_P   17.3938]*AlbCon         0
0         0         0    2    130       -130          20     1.5  0.5; %Gen
34 bus
  119    1.0127   21.9026         0         0   -0.0000   -0.0000         0
23.2500*AlbCon    3    0.0        0.0         500    1.5  0.5; %HV bus for
Gen 34 and load bus 120, Alberta
  120    0.9821   16.1324         0         0   [93.0000-11   23.2500-
4]*AlbCon         0         0    3    0.0        0.0         110    1.5  0.5;
%Load bus
  121    1.0140  -41.6678         0         0    0.0000    0.0000         0
0    3    0.0        0.0         500    1.5  0.5; %TCSC bus 1
  122    1.0140  -41.6678         0         0    0.0000    0.0000         0
0    3    0.0        0.0         500    1.5  0.5];%TCSC bus 2

%% Line data
AlbCon2 = 0.01;
if AlbCon; AlbCon2 = 1; end
line = [
%         Fbus         Tbus     R          X                    Y
            1            2            0    0.0026667            0; %Gen 1
stepup x-former
            3            4            0    0.0032432            0; %Gen 2
stepup x-former
            5            6            0    0.0032432            0; %Gen 3
stepup x-former
            9           10            0        0.006            0; %Gen 4
stepup x-former
           12           13            0        0.003            0; %Gen 5
stepup x-former
           14           15            0         0.01            0; %Gen 6
stepup x-former
           17           18            0    0.0014458            0; %Gen 7
stepup x-former
           19           20            0    0.0033333            0; %Gen 8
stepup x-former
           22           20            0    0.0097561            0; %Gen 9
stepup x-former
           23           24            0       0.0024            0; %Gen 10
stepup x-former
           25           24            0        0.004            0; %Gen 11
stepup x-former
           27           28            0         0.04            0; %Gen 12
stepup x-former
           30           31            0    0.0066667            0; %Gen 13
stepup x-former
           32           33            0    0.0052174            0; %Gen 14
stepup x-former
           34           35            0     0.018462            0; %Gen 15
stepup x-former
           37           38            0        0.004            0; %Gen 16
stepup x-former 1 (seel below for x-former 2)
```

```
           39           44            0       0.0048            0; %Gen 17
stepup x-former
           41           42            0    0.0011538            0; %Gen 18
stepup x-former
           45           46            0    0.0034286            0; %Gen 19
stepup x-former
           47          115            0    0.0046154            0; %Gen 20
stepup x-former
           48           49            0   0.00067416            0; %Gen 21
stepup x-former
           51           52            0       0.0048            0; %Gen 22
stepup x-former
           53           54            0    0.0034286            0; %Gen 23
stepup x-former
           58           57            0      0.00375            0; %Gen 24
stepup x-former
           59           57            0    0.0023077            0; %Gen 25
stepup x-former
           60           61            0    0.0025532            0; %Gen 26
stepup x-former
           62           63            0    0.0010619            0; %Gen 27
stepup x-former
           65           66            0       0.0012            0; %Gen 28
stepup x-former
           68           69            0    0.0011429            0; %Gen 29
stepup x-former
           71           72            0    0.0012371            0; %Gen 30
stepup x-former
           74           75            0        0.005            0; %Gen 31
stepup x-former
           76           77            0    0.0054545            0; %Gen 32
stepup x-former
          107          108            0         0.01            0; %Gen 33
stepup x-former
          118          119            0   0.00092308            0; %Gen 34
stepup x-former
            8            7            0      0.00125            0; %Load bus
8 x-former
           11           10            0    0.0016667            0; %Load bus
11 x-former
           16           15            0       0.0025            0; %Load bus
16 x-former
           21           20            0         0.01            0; %Load bus
21 x-former
           26           24            0         0.02            0; %Load bus
26 x-former
           29           28            0    0.0057143            0; %Load bus
29 x-former
           36           35            0         0.01            0; %Load bus
36 x-former
           43           44            0   0.00043103            0; %Load bus
43 x-former
           50           49            0   0.00076923            0; %Load bus
50 x-former
           55           54            0     0.002439            0; %Load bus
55 x-former
```

```
           56           57            0      0.00125            0; %Load bus
56 x-former
           64           63            0   0.00074627            0; %Load bus
64 x-former
           95           67            0    0.0033333            0; %Load bus
95 x-former
           70           69            0    0.0015385            0; %Load bus
70 x-former
           73           72            0    0.0011111            0; %Load bus
73 x-former
           78           77            0    0.0033333            0; %Load bus
78 x-former
          109          108            0        0.005            0; %Load bus
109 x-former
          112          113            0   0.00064516            0; %Load bus
112 x-former
          120          119            0    0.0010753            0; %Load bus
120 x-former
            2           80     0.001679     0.012916            0
           80           79       0.0014       0.0095            0
           80           81      0.00145       0.0117            0
           81            7      0.00145       0.0117            0
           80            7       0.0033       0.0221            0
            4           79     0.000309     0.004232            0
            6           79      0.00265       0.0366            0
            6           79       0.0028       0.0396            0
           79            7      0.00125      0.00825            0
           79            7       0.0025       0.0184            0
            6           28     0.003125       0.0375            0; %BC-North
(includes x-former)
            7           10     0.000746     0.012954            0
           18           10       0.0013       0.0251            0
           18           82       0.0004       0.0063            0
           82           10     0.000203     0.003377            0
           10           13      0.00075        0.018            0
           13           15      0.00075        0.018            0
           82           24     0.000343     0.007409            0
           18           28      0.00064      0.01456            0
           33           83       0.0035       0.0438            0
           33           85       0.0028      0.03504            0
           83           85       0.0007      0.00876            0
           83           84       0.0012       0.0222            0
           83           84       0.0012       0.0222            0
           31           84       0.0007       0.0124            0
           84           20      0.00215      0.03135            0
           84           28       0.0012       0.0202            0
           18           20       0.0011       0.0207            0
           20           24            0        0.002            0
           24           15      0.00025        0.006            0
           24           86       0.0007       0.0112            0
           24           86       0.0011       0.0211            0
           24           86       0.0011       0.0211            0
           86           87     0.001075     0.004175            0
           87          121  0.001075     0.004175            0; %Added TCSC
           86           88       0.0012       0.0034            0
           86          121    0.0022       0.0084            0; %Added TCSC
           88          121    0.0008       0.0076            0; %Added TCSC
```

```
           88          105        0.0039         0.024             0
           89           35        0.0017        0.0261             0
           15           35        0.0014        0.0136             0
           77          102         0.005          0.02             0
          105          104             0          0.01             0; %x-former,
North
           83          103             0          0.01             0; %x-former,
North
          103          102          0.03          0.12             0
          102          104             0         0.002             0; %x-former,
North
          104          101        0.0025      0.008333             0
          104          101        0.0025      0.008333             0
          101           75        0.0025         0.015             0
           20          111             0          0.01             0; %x-former,
North
          111           77          0.02          0.24             0
          101           69         0.002          0.02             0
          101          100         0.001          0.01             0
          100           69         0.001          0.01             0
          104          108       0.01275      0.041667             0
           89           38         0.001         0.009             0
           89           38        0.0011        0.0089             0
           89           90        0.0011        0.0123             0
           89          110             0          0.01             0; %x-former,
North-West
          110          108          0.02      0.066667             0
           38           40        0.0011        0.0059             0
           38           40        0.0011        0.0059             0
           40           44        0.0014        0.0117             0
           40           44        0.0015        0.0102             0
           90           44        0.0021        0.0124             0
           90           44         0.002         0.012             0
           42           44             0        0.0003             0
           44          114        0.0007        0.0159             0
           44          114        0.0006        0.0141             0
           44          114        0.0015        0.0307             0
          114           46        0.0015        0.0145             0
          114           99        0.0008         0.009             0
           99          115        0.0008        0.0202             0
           99           46        0.0007        0.0055             0
          115           46       0.00045       0.01055             0
           46           49        0.0011        0.0082             0
           46           49        0.0011         0.008             0
           46           91       0.00055       0.00335             0
           91           49       0.00055       0.00335             0
           49          113        0.0002       0.00565             0
           49           52         0.002          0.02             0; %West
(includes x-former)
           52           54         0.002          0.02             0; %West
(includes x-former)
          113           57        0.0008        0.0126             0
          113           57        0.0012        0.0226             0
          113           57        0.0017        0.0128             0
          113          116       0.00115       0.01715             0
          116           57       0.00115       0.01715             0
```

```
           57           94            0         0.01            0; %West-East
(includes x-former)
           57           66       0.0021       0.0143            0
           57           98     0.000512     0.003589            0
           98           66     0.000512     0.003589            0
           57           63       0.0085        0.095            0
           57           93       0.0009       0.0082            0
           93           63       0.0009       0.0082            0
          113           92    0.0010875     0.016819            0
           92           61    0.0010875     0.016819            0
           54           61       0.0021     0.024473            0
           61           63            0         0.01            0
           94           69        0.015         0.05            0
           63           66       0.0018       0.0275            0
           63           97      0.00135        0.016            0
           97           66      0.00135        0.016            0
           66           67       0.0009      0.02085            0
           67           96            0        0.003            0; %x-former,
East
           63           96      0.00375       0.0125            0
           96           69       0.0075        0.025            0
           96           72      0.01125       0.0375            0
           72           69     0.014062     0.046875            0
          108           69         0.02     0.066667            0
            6          106       0.0046*AlbCon       0.0641*AlbCon2
0
          106          119            0*AlbCon       0.02*AlbCon2
0
          106          117            0*AlbCon       0.01*AlbCon2
0
          117          119            0*AlbCon       0.01*AlbCon2
0
           37           90            0        0.004            0; %Gen 16
stepup x-former 2
           52          113       0.0002      0.00565            0
           63           66        0.004        0.052            0;
          121          122            0         1/YTCSCf        0; %Inductor
to make TCSC net impedance zero
          122           89            0        -1/YTCSCf        0]; %TCSC
fixed capacitance

% Machine data format
%   Column
%     1. machine number (may be different from bus number),
%     2. bus number,
%     3. machine base mva,
%     4. leakage reactance x_l(pu),
%     5. resistance r_a(pu),
%     6. d-axis sychronous reactance x_d(pu),
%     7. d-axis transient reactance x'_d(pu),
%     8. d-axis subtransient reactance x"_d(pu),
%     9. d-axis open-circuit time constant T'_do(sec),
%    10. d-axis open-circuit subtransient time constant T"_do(sec),
%    11. q-axis sychronous reactance x_q(pu),
%    12. q-axis transient reactance x'_q(pu),
%    13. q-axis subtransient reactance x"_q(pu),
%        NOTE:  PST requires that x"_q = x"_d
```

```
%    14. q-axis open-circuit time constant T'_qo(sec),
%        if T'_q0=0, PST sets it to 999 (i.e., removes from model),
%    15. q-axis open circuit subtransient time constant T"_qo(sec),
%        if T"_q0=0, PST sets it to 999,
%    16. inertia constant H(sec),
%    17. damping coefficient d_o(pu),
%    18. dampling coefficient d_1(pu), (relative to pmech)
%    19. bus number
%    20. s(1.0)
%    21. s(1.2)
% John Undrill reccomended the x'_q, T'_qo, and T"_qo values for the
% salient machines.
d_0S = 0; %Salient damping
d_0R = 0; %Round rotor damping
d_1 = 1; %Pmech damping (Not used in new mac_sub)
mac_con = [
% 1   2   3      4x    5y   6y   7y    8y    9y     10y    11y   12y   13y
14y   15y    16y   17    18   19  20x    21x
% num bus base   x_l   r_a  x_d  x'_d  x"_d  T'_do  T"_do  x_q   x'_q  x"_q
T'_qo T"_qo  H     d_0   d_1  bus s(1.0) s(1.2)
   1   1  4500   0.17  0.0  1.2  0.3   0.22  6.0    0.025  0.7   0.23  0.22
0.06  0.04   5.0   d_0S  d_1   1  0.05   0.3; %Hydro, salient pole
   2   3  3700   0.17  0.0  1.2  0.3   0.22  6.0    0.025  0.7   0.23  0.22
0.06  0.04   5.0   d_0S  d_1   3  0.05   0.3; %Hydro, salient pole
   3   5  3700   0.17  0.0  1.2  0.3   0.22  6.0    0.025  0.7   0.23  0.22
0.06  0.04   5.0   d_0S  d_1   5  0.05   0.3; %Hydro, salient pole
   4   9  2000   0.17  0.0  2.0  0.2   0.18  7.5    0.025  1.9   0.7   0.18
0.5   0.06   6.5   d_0R  d_1   9  0.05   0.3; %Gas Turbine, round rotor
   5  12  4000   0.17  0.0  2.0  0.2   0.18  7.5    0.025  1.9   0.7   0.18
0.5   0.06   6.5   d_0R  d_1  12  0.05   0.3; %Steam, round rotor
   6  14  1200   0.17  0.0  2.0  0.2   0.18  7.5    0.025  1.9   0.7   0.18
0.5   0.06   6.5   d_0R  d_1  14  0.05   0.3; %Gas Turbine, round rotor
   7  17  8300   0.17  0.0  1.2  0.3   0.22  6.0    0.025  0.7   0.23  0.22
0.06  0.04   5.0   d_0S  d_1  17  0.05   0.3; %Hydro, salient pole
   8  19  3600   0.17  0.0  1.2  0.3   0.22  6.0    0.025  0.7   0.23  0.22
0.06  0.04   5.0   d_0S  d_1  19  0.05   0.3; %Hydro, salient pole
   9  22  1230   0.17  0.0  2.0  0.2   0.18  7.5    0.025  1.9   0.7   0.18
0.5   0.06   6.5   d_0R  d_1  22  0.05   0.3; %Steam, round rotor
  10  23  5000   0.17  0.0  1.2  0.3   0.22  6.0    0.025  0.7   0.23  0.22
0.06  0.04   5.0   d_0S  d_1  23  0.05   0.3; %Hydro, salient pole
  11  25  3000   0.17  0.0  2.0  0.2   0.18  7.5    0.025  1.9   0.7   0.18
0.5   0.06   6.5   d_0R  d_1  25  0.05   0.3; %Gas Turbine, round rotor
  12  27   300   0.17  0.0  2.0  0.2   0.18  7.5    0.025  1.9   0.7   0.18
0.5   0.06   6.5   d_0R  d_1  27  0.05   0.3; %Gas Turbine, round rotor
  13  30  1800   0.17  0.0  1.2  0.3   0.22  6.0    0.025  0.7   0.23  0.22
0.06  0.04   5.0   d_0S  d_1  30  0.05   0.3; %Hydro, salient pole
  14  32  2300   0.17  0.0  2.0  0.2   0.18  7.5    0.025  1.9   0.7   0.18
0.5   0.06   6.5   d_0R  d_1  32  0.05   0.3; %Steam, round rotor
  15  34   650   0.17  0.0  2.0  0.2   0.18  7.5    0.025  1.9   0.7   0.18
0.5   0.06   6.5   d_0R  d_1  34  0.05   0.3; %Gas Turbine, round rotor
  16  37  3000   0.17  0.0  1.2  0.3   0.22  6.0    0.025  0.7   0.23  0.22
0.06  0.04   5.0   d_0S  d_1  37  0.05   0.3; %Hydro, salient pole
  17  39  2500   0.17  0.0  2.0  0.2   0.18  7.5    0.025  1.9   0.7   0.18
0.5   0.06   6.5   d_0R  d_1  39  0.05   0.3; %Gas Turbine, round rotor
  18  41  5200   0.17  0.0  2.0  0.2   0.18  7.5    0.025  1.9   0.7   0.18
0.5   0.06   6.5   d_0R  d_1  41  0.05   0.3; %Gas Turbine, round rotor
```

```
  19  45  3500   0.17  0.0  2.0  0.2   0.18  7.5    0.025  1.9   0.7   0.18
0.5   0.06   6.5   d_0R  d_1  45  0.05   0.3; %Gas Turbine, round rotor
  20  47  2600   0.17  0.0  2.0  0.2   0.18  7.5    0.025  1.9   0.7   0.18
0.5   0.06   6.5   d_0R  d_1  47  0.05   0.3; %Steam, round rotor
  21  48 17800   0.17  0.0  2.0  0.2   0.18  7.5    0.025  1.9   0.7   0.18
0.5   0.06   6.5   d_0R  d_1  48  0.05   0.3; %Steam, round rotor? LA
  22  51  2500   0.17  0.0  2.0  0.2   0.18  7.5    0.025  1.9   0.7   0.18
0.5   0.06   6.5   d_0R  d_1  51  0.05   0.3; %Steam, round rotor
  23  53  3500   0.17  0.0  2.0  0.2   0.18  7.5    0.025  1.9   0.7   0.18
0.5   0.06   6.5   d_0R  d_1  53  0.05   0.3; %Steam, round rotor? SD
  24  58  3200   0.17  0.0  1.2  0.3   0.22  6.0    0.025  0.7   0.23  0.22
0.06  0.04   5.0   d_0S  d_1  58  0.05   0.3; %Hydro, salient pole
  25  59  5200   0.17  0.0  2.0  0.2   0.18  7.5    0.025  1.9   0.7   0.18
0.5   0.06   6.5   d_0R  d_1  59  0.05   0.3; %Gas Turbine, round rotor
  26  60  4700   0.17  0.0  2.0  0.2   0.18  7.5    0.025  1.9   0.7   0.18
0.5   0.06   6.5   d_0R  d_1  60  0.05   0.3; %Steam, round rotor
  27  62 11300   0.17  0.0  2.0  0.2   0.18  7.5    0.025  1.9   0.7   0.18
0.5   0.06   6.5   d_0R  d_1  62  0.05   0.3; %Gas Turbine, round rotor
  28  65 10000   0.17  0.0  2.0  0.2   0.18  7.5    0.025  1.9   0.7   0.18
0.5   0.06   6.5   d_0R  d_1  65  0.05   0.3; %Steam, round rotor
  29  68 10500   0.17  0.0  2.0  0.2   0.18  7.5    0.025  1.9   0.7   0.18
0.5   0.06   6.5   d_0R  d_1  68  0.05   0.3; %Steam, round rotor
  30  71  9700   0.17  0.0  2.0  0.2   0.18  7.5    0.025  1.9   0.7   0.18
0.5   0.06   6.5   d_0R  d_1  71  0.05   0.3; %Steam, round rotor
  31  74  2400   0.17  0.0  2.0  0.2   0.18  7.5    0.025  1.9   0.7   0.18
0.5   0.06   6.5   d_0R  d_1  74  0.05   0.3; %Steam, round rotor
  32  76  2200   0.17  0.0  1.2  0.3   0.22  6.0    0.025  0.7   0.23  0.22
0.06  0.04   5.0   d_0S  d_1  76  0.05   0.3; %Hydro, salient pole
  33 107  1200   0.17  0.0  2.0  0.2   0.18  7.5    0.025  1.9   0.7   0.18
0.5   0.06   6.5   d_0R  d_1 107  0.05   0.3]; %Steam, round rotor? Reno

if AlbCon
    mac_con = [mac_con;
  34 118 13000   0.17  0.0  2.0  0.2   0.18  7.5    0.025  1.9   0.7   0.18
0.5   0.06   6.5   d_0R  d_1 118  0.05   0.3]; %Steam, round rotor
end

% exciter model
% exc_con matrix format
%column        data
%  1    exciter type (=3 for exc_st3, IEEE type ST3)
%  2    machine number
%  3    transducer filter time constant (T_R - sec)
%  4    voltage regulator gain (K_A)
%  5    voltage regulator time constant (T_A - sec)
%  6    transient gain reduction time constnat (T_B - sec) -- denominator
%  7    transient gain reduction time constnat (T_C - sec) -- numerator
%  8    max voltage regulator output (V_Rmax - pu)
%  9    min voltage regulator output (V_Rmin - pu)
% 10    max internal signal (VImax - pu)
% 11    min internal signal (VImin - pu)
% 12    first state regulator gain (KJ)
% 13    potential circuit gain coef (KP)
% 14    potential circuit phase angle (qP - degrees)
% 15    current circuit gain coef (KI)
% 16    potential source reactance (XL - pu)
```

```
% 17    rectifier loading factor (KC)
% 18    max field voltage (Efdmax - pu)
% 19    inner loop feedback constant (KG)
% 20    max innerloop voltage feedback (VGmax - pu)
exc_con = [...
%   1   2   3     4     5     6     7    8     9     10    11    12  13  14
15  16  17 18?    19  20
%  type num T_R   K_A   T_A   T_B   T_C  Vrmax Vrmin VImax VImin KJ  KP  qP
KI  XL  KC Efdmax KG  VGmax
    3    1  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3    2  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3    3  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3    4  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3    5  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3    6  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3    7  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3    8  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3    9  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3   10  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3   11  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3   12  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3   13  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3   14  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3   15  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3   16  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3   17  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3   18  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3   19  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3   20  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3   21  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3   22  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3   23  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3   24  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
```

```
    3   25  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3   26  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3   27  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3   28  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3   29  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3   30  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3   31  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3   32  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3   33  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0];

if AlbCon
exc_con = [exc_con;
    3   34  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0];
end

% PSS model
% pss_con matrix format
%column        data         unit
%  1      Type input (1=spd, 2=power)
%  2      machine number
%  3      gain K
%  4      Washout time const Tw (sec)
%  5      1st lead time const T1 (sec)
%  6      1st lag time const T2 (sec)
%  7      2nd lead time const T3 (sec)
%  8      2nd lag time const T4 (sec)
%  9      max output limit (pu)
%  10     min output limit (pu)
pss_con = [];
for k=1:length(pssGen)
%                    type gen#      K  Tw T1   T2   T3   T4    max min
    pss_con(k,:) = [1    pssGen(k) 20 2  0.25 0.04 0.2  0.03  0.1 -0.1];
end

%Gens 1, 2, 13, 14, and 26,

% governor model
% tg_con matrix format
%column        data         unit
%  1    turbine model number (=1)
%  2    machine number
%  3    speed set point   wf        pu
%  4    steady state gain 1/R       pu
%  5    maximum power order  Tmax   pu on generator base
%  6    servo time constant   Ts    sec
%  7    governor time constant  Tc  sec
```

```
%  8    transient gain time constant T3 sec
%  9    HP section time constant   T4   sec
% 10    reheater time constant    T5    sec
tg_con = [...
%     num  wf 1/R        Tmax  Ts    Tc   T3     T4     T5
1     1   1  20         1.0   0.50  15   1.0    -1.0   0.5;%Slow Hydro
1     2   1  20         1.0   0.50  15   1.0    -1.0   0.5;%Slow Hydro
1     3   1  20         1.0   0.50  5    1.0    -1.0   0.5;%Fast Hydro
%1    4   1  20         1.0   0.50  10   4.0    0      1.0;%Gas Turbine - NO
GOV
%1    5   1  20         1.0   0.50  10   3.0    0      0.05;%Steam - NO GOV
%1    6   1  20         1.0   0.50  10   4.0    0      1.0;%Gas Turbine - NO
GOV
1     7   1  20         1.0   0.50  15   1.0    -1.0   0.5;%Slow Hydro
1     8   1  20         1.0   0.50  15   1.0    -1.0   0.5;%Slow Hydro
%1    9   1  20         1.0   0.10  10   3.0    0      0.01;%Steam - NO GOV
1    10   1  20         1.0   0.50  15   1.0    -1.0   0.5;%Slow Hydro
%1   11   1  20         1.0   0.50  10   4.0    0      1.0;%Gas Turbine - NO
GOV
%1   12   1  20         1.0   0.50  10   4.0    0      1.0;%Gas Turbine - NO
GOV
1    13   1  20         1.0   0.50  5    1.0    -1.0   0.5;%Fast Hydro
1    14   1  20         1.0   0.10  10   3.0    0      0.01;%Steam
%1   15   1  20         1.0   0.50  10   4.0    0      1.0;%Gas Turbine - NO
GOV
1    16   1  20         1.0   0.50  5    1.0    -1.0   0.5;%Fast Hydro
%1   17   1  20         1.0   0.10  10   3.0    0      0.01;%Steam - NO GOV
1    18   1  20*0.25    1.0   0.50  10   4.0    0      1.0;%Gas Turbine - 75%
BASE LOADED
1    19   1  20*0.25    1.0   0.50  10   4.0    0      1.0;%Gas Turbine - 75%
BASE LOADED
%1   20   1  20         1.0   0.10  10   3.0    0      0.01;%Steam - NO GOV
1    21   1  20         1.0   0.10  10   3.0    0      0.01;%Steam
%1   22   1  20         1.0   0.10  10   3.0    0      0.01;%Steam - NO GOV
1    23   1  20         1.0   0.10  10   3.0    0      0.01;%Steam
1    24   1  20         1.0   0.50  15   1.0    -1.0   0.5;%Slow Hydro
1    25   1  20*0.25    1.0   0.50  10   4.0    0      1.0;%Gas Turbine
%1   26   1  20         1.0   0.10  10   3.0    0      0.01;%Steam - NO GOV
1    27   1  20*0.25    1.0   0.50  10   4.0    0      1.0;%Gas Turbine
1    28   1  20*0.2     1.0   0.10  10   3.0    0      0.01;%Steam
%1   29   1  20         1.0   0.10  10   3.0    0      0.01;%Steam - NO GOV
1    30   1  20*0.25    1.0   0.10  10   3.0    0      0.01;%Steam
%1   31   1  20         1.0   0.10  10   3.0    0      0.01;%Steam - NO GOV
1    32   1  20         1.0   0.50  5    1.0    -1.0   0.5;%Fast Hydro
1    33   1  20         1.0   0.10  10   3.0    0      0.01];%Steam

if AlbCon
    tg_con = [tg_con;
1    34   1  20*0.25    1.0   0.10  10   3.0    0      0.01];%Steam
end

% non-conforming load
% col 1       bus number
% col 2       fraction const active power load
% col 3       fraction const reactive power load
% col 4       fraction const active current load
```

```
% col 5       fraction const reactive current load
load_con = [...
%bus Pcont Qconst P_Icont Q_Iconst
  8   0     0     1       0; %Load
 11   0     0     1       0; %Load
 16   0     0     1       0; %Load
 21   0     0     1       0; %Load
 24   0     0     1       0; %PDCI North
 26   0     0     1       0; %Load
 29   0     0     1       0; %Load
 36   0     0     1       0; %Load
 43   0     0     1       0; %Load
 49   0     0     1       0; %PDCI South
 50   0     0     1       0; %Load
 55   0     0     1       0; %Load
 56   0     0     1       0; %Load
 64   0     0     1       0; %Load
 69   0     0     1       0; %DC North
 70   0     0     1       0; %Load
 73   0     0     1       0; %Load
 78   0     0     1       0; %Load
 95   0     0     1       0; %Load
109   0     0     1       0; %Load
112   0     0     1       0; %Load
113   0     0     1       0; %DC South
120   0     0     1       0; %Load
  2   0     0     1       0; %BC modulation
  4   0     0     1       0; %BC modulation
  6   0     0     1       0; %BC modulation
  7   0     0     1       0; %BC modulation
 79   0     0     1       0; %BC modulation
 80   0     0     1       0; %BC modulation
 10   0     0     1       0; %North modulation
 13   0     0     1       0; %North modulation
 15   0     0     1       0; %North modulation
 18   0     0     1       0; %North modulation
 20   0     0     1       0; %North modulation
 28   0     0     1       0; %North modulation
 31   0     0     1       0; %North modulation
 33   0     0     1       0; %North modulation
 35   0     0     1       0; %North modulation
 75   0     0     1       0; %North modulation
 77   0     0     1       0; %North modulation
 83   0     0     1       0; %North modulation
 88   0     0     1       0; %North modulation
 89   0     0     1       0; %North modulation
101   0     0     1       0; %North modulation
 38   0     0     1       0; %nWest modulation
 40   0     0     1       0; %nWest modulation
 42   0     0     1       0; %nWest modulation
 44   0     0     1       0; %nWest modulation
 90   0     0     1       0; %nWest modulation
108   0     0     1       0; %nWest modulation
114   0     0     1       0; %nWest modulation
 46   0     0     1       0; %sWest modulation
 52   0     0     1       0; %sWest modulation
 54   0     0     1       0; %sWest modulation
```

```
 57   0     0     1       0; %sWest modulation
 99   0     0     1       0; %sWest modulation
115   0     0     1       0; %sWest modulation
 61   0     0     1       0; %East modulation
 63   0     0     1       0; %East modulation
 66   0     0     1       0; %East modulation
 67   0     0     1       0; %East modulation
 72   0     0     1       0; %East modulation
106   0     0     1       0; %Alberta modulation
119   0     0     1       0; %Alberta modulation
121   0     0     0       0; %TCSC bus
122   0     0     0       0]; %TCSC bus

% Load modulation data (sets up modulation of real part of load)
% col 1       load modulation number (Index for b_lmod and lmod_sig)
% col 2       bus number
% col 3       modulation base MVA (MVA)
% col 4       max conductance (pu)
% col 5       min conductance (pu)
% col 6       regulator gain (K)
% col 7       regulator time constant (TR)
% NOTE: This creates b_lmod for the linear analysis.
lmod_con=[...
%num bus  MVA  Max  Min  K  TR
  1   2   100  100 -100  1  0.05; %BC
  2   4   100  100 -100  1  0.05; %BC
  3   6   100  100 -100  1  0.05; %BC
  4   7   100  100 -100  1  0.05; %BC
  5  79   100  100 -100  1  0.05; %BC
  6  80   100  100 -100  1  0.05; %BC
  7  10   100  100 -100  1  0.05; %North
  8  13   100  100 -100  1  0.05; %North
  9  15   100  100 -100  1  0.05; %North
 10  18   100  100 -100  1  0.05; %North
 11  20   100  100 -100  1  0.05; %North
 12  24   100  100 -100  1  0.05; %North PDCI
 13  28   100  100 -100  1  0.05; %North
 14  31   100  100 -100  1  0.05; %North
 15  33   100  100 -100  1  0.05; %North
 16  35   100  100 -100  1  0.05; %North
 17  75   100  100 -100  1  0.05; %North
 18  77   100  100 -100  1  0.05; %North
 19  83   100  100 -100  1  0.05; %North
 20  88   100  100 -100  1  0.05; %North
 21  89   100  100 -100  1  0.05; %North
 22 101   100  100 -100  1  0.05; %North
 23  38   100  100 -100  1  0.05; %nWest
 24  40   100  100 -100  1  0.05; %nWest
 25  42   100  100 -100  1  0.05; %nWest
 26  44   100  100 -100  1  0.05; %nWest
 27  90   100  100 -100  1  0.05; %nWest
 28 108   100  100 -100  1  0.05; %nWest
 29 114   100  100 -100  1  0.05; %nWest
 30  46   100  100 -100  1  0.05; %sWest
 31  49   100  100 -100  1  0.05; %sWest PDCI
 32  52   100  100 -100  1  0.05; %sWest
 33  54   100  100 -100  1  0.05; %sWest
```

```
 34  57   100  100 -100  1  0.05; %sWest
 35  99   100  100 -100  1  0.05; %sWest
 36 113   100  100 -100  1  0.05; %sWest DC
 37 115   100  100 -100  1  0.05; %sWest
 38  61   100  100 -100  1  0.05; %East
 39  63   100  100 -100  1  0.05; %East
 40  66   100  100 -100  1  0.05; %East
 41  67   100  100 -100  1  0.05; %East
 42  69   100  100 -100  1  0.05; %East DC
 43  72   100  100 -100  1  0.05; %East
 44 106   100  100 -100  1  0.05; %Alberta
 45 119   100  100 -100  1  0.05; %Alberta
 46   8   100  100 -100  bus(8,6)/100   0.05; %Load
 47  11   100  100 -100  bus(11,6)/100  0.05; %Load
 48  16   100  100 -100  bus(16,6)/100  0.05; %Load
 49  21   100  100 -100  bus(21,6)/100  0.05; %Load
 50  26   100  100 -100  bus(26,6)/100  0.05; %Load
 51  29   100  100 -100  bus(29,6)/100  0.05; %Load
 52  36   100  100 -100  bus(36,6)/100  0.05; %Load
 53  43   100  100 -100  bus(43,6)/100  0.05; %Load
 54  50   100  100 -100  bus(50,6)/100  0.05; %Load
 55  55   100  100 -100  bus(55,6)/100  0.05; %Load
 56  56   100  100 -100  bus(56,6)/100  0.05; %Load
 57  64   100  100 -100  bus(64,6)/100  0.05; %Load
 58  70   100  100 -100  bus(70,6)/100  0.05; %Load
 59  73   100  100 -100  bus(73,6)/100  0.05; %Load
 60  78   100  100 -100  bus(78,6)/100  0.05; %Load
 61  95   100  100 -100  bus(95,6)/100  0.05; %Load
 62 109   100  100 -100  bus(109,6)/100 0.05; %Load
 63 112   100  100 -100  bus(112,6)/100 0.05; %Load
 64 120   100  100 -100  bus(120,6)/100 0.05]; %Load

% Reactive load modulation matrix.
% col 1       load modulation number (Index for b_rlmod and rlmod_sig)
% col 2       bus number
% col 3       modulation base MVA (MVA)
% col 4       max conductance (pu)
% col 5       min conductance (pu)
% col 6       regulator gain (K)
% col 7       regulator time constant (TR)
% NOTE: This creates b_rlmod for the linear analysis.
rlmod_con=[...
%num bus  MVA  Max  Min  K  TR
  1   2   100  100 -100  1  0.05; %BC
  2   4   100  100 -100  1  0.05; %BC
  3   6   100  100 -100  1  0.05; %BC
  4   7   100  100 -100  1  0.05; %BC
  5  79   100  100 -100  1  0.05; %BC
  6  80   100  100 -100  1  0.05; %BC
  7  10   100  100 -100  1  0.05; %North
  8  13   100  100 -100  1  0.05; %North
  9  15   100  100 -100  1  0.05; %North
 10  18   100  100 -100  1  0.05; %North
 11  20   100  100 -100  1  0.05; %North
 12  24   100  100 -100  1  0.05; %North PDCI
 13  28   100  100 -100  1  0.05; %North
 14  31   100  100 -100  1  0.05; %North
```

```
 15  33   100  100 -100  1  0.05; %North
 16  35   100  100 -100  1  0.05; %North
 17  75   100  100 -100  1  0.05; %North
 18  77   100  100 -100  1  0.05; %North
 19  83   100  100 -100  1  0.05; %North
 20  88   100  100 -100  1  0.05; %North
 21  89   100  100 -100  1  0.05; %North
 22 101   100  100 -100  1  0.05; %North
 23  38   100  100 -100  1  0.05; %nWest
 24  40   100  100 -100  1  0.05; %nWest
 25  42   100  100 -100  1  0.05; %nWest
 26  44   100  100 -100  1  0.05; %nWest
 27  90   100  100 -100  1  0.05; %nWest
 28 108   100  100 -100  1  0.05; %nWest
 29 114   100  100 -100  1  0.05; %nWest
 30  46   100  100 -100  1  0.05; %sWest
 31  49   100  100 -100  1  0.05; %sWest PDCI
 32  52   100  100 -100  1  0.05; %sWest
 33  54   100  100 -100  1  0.05; %sWest
 34  57   100  100 -100  1  0.05; %sWest
 35  99   100  100 -100  1  0.05; %sWest
 36 113   100  100 -100  1  0.05; %sWest DC
 37 115   100  100 -100  1  0.05; %sWest
 38  61   100  100 -100  1  0.05; %East
 39  63   100  100 -100  1  0.05; %East
 40  66   100  100 -100  1  0.05; %East
 41  67   100  100 -100  1  0.05; %East
 42  69   100  100 -100  1  0.05; %East DC
 43  72   100  100 -100  1  0.05; %East
 44 106   100  100 -100  1  0.05; %Alberta
 45 119   100  100 -100  1  0.05; %Alberta
 46   8   100  100 -100  bus(8,7)/100   0.05; %Load
 47  11   100  100 -100  bus(11,7)/100  0.05; %Load
 48  16   100  100 -100  bus(16,7)/100  0.05; %Load
 49  21   100  100 -100  bus(21,7)/100  0.05; %Load
 50  26   100  100 -100  bus(26,7)/100  0.05; %Load
 51  29   100  100 -100  bus(29,7)/100  0.05; %Load
 52  36   100  100 -100  bus(36,7)/100  0.05; %Load
 53  43   100  100 -100  bus(43,7)/100  0.05; %Load
 54  50   100  100 -100  bus(50,7)/100  0.05; %Load
 55  55   100  100 -100  bus(55,7)/100  0.05; %Load
 56  56   100  100 -100  bus(56,7)/100  0.05; %Load
 57  64   100  100 -100  bus(64,7)/100  0.05; %Load
 58  70   100  100 -100  bus(70,7)/100  0.05; %Load
 59  73   100  100 -100  bus(73,7)/100  0.05; %Load
 60  78   100  100 -100  bus(78,7)/100  0.05; %Load
 61  95   100  100 -100  bus(95,7)/100  0.05; %Load
 62 109   100  100 -100  bus(109,7)/100 0.05; %Load
 63 112   100  100 -100  bus(112,7)/100 0.05; %Load
 64 120   100  100 -100  bus(120,7)/100 0.05]; %Load

% Monitored lines.  When conducting an eigenanalysis, this causes
% c_pf1, c_pf2, c_qf1, c_qf2, c_ilif, c_ilit, c_ilmf, c_ilmt, c_ilrf,
% c_ilrt to be created.  The ith row of c_pf1
% and c_qf1 correspond to line(lmon_con(i),:) of the line matrix.
lmon_con = [1:size(line,1)]; %All lines
```

22. Appendix M: Switching Logic for Local Fault and Remote Load-Shed Cases

Local Fault:

```
%% Switching
%Switching file defines the simulation control
% row 1 col1  simulation start time (s) (cols 2 to 6 zeros)
%     col7  initial time step (s)
% row 2 col1  fault application time (s)
%     col2  bus number at which fault is applied
%     col3  bus number defining far end of faulted line
%     col4  zero sequence impedance in pu on system base
%     col5  negative sequence impedance in pu on system base
%     col6  type of fault  - 0 three phase
%                  - 1 line to ground
%                  - 2 line-to-line to ground
%                  - 3 line-to-line
%                  - 4 loss of line with no fault
%                  - 5 loss of load at bus
%     col7  time step for fault period (s)
% row 3 col1  near end fault clearing time (s) (cols 2 to 6 zeros)
%     col7  time step for second part of fault (s)
% row 4 col1  far end fault clearing time (s) (cols 2 to 6 zeros)
%     col7  time step for fault cleared simulation (s)
% row 5 col1  time to change step length (s)
%     col7  time step (s)
%
sw_con = [...
 0           0    0    0    0    0    1/240; %sets intitial time step
 1.0        83   84    0    0    0    1/240; %apply fault (three phase)
 1+10/60     0    0    0    0    0    1/240; %clear near end of fault
 1+11/60     0    0    0    0    0    1/240; %clear far end of fault
 3           0    0    0    0    0    1/60; %increas time step
 5           0    0    0    0    0    0]; %end simulation
```

Remote Load-Shed:

```
sw_con = [...
 0           0    0    0    0    0    1/240; %sets intitial time step
 1.0         8    7    0    0    0    1/240; %apply fault (three phase)
 1+9/60      0    0    0    0    0    1/240; %clear near end of fault
 1+10/60     0    0    0    0    0    1/240; %clear far end of fault
 5           0    0    0    0    0     1/60; %increase time step
 10          0    0    0    0    0       0]; %end simulation%Line 6-106
```

23. Appendix N: MicroWECC V1 PST Data File

```
% MicroWECC V1
% Based on d_minniWECC_V3C_C3_6.m
% PST data file
%
%
% System bases:  100 MVA, 60 Hz.

% Written by RJ Hallett, 2017

pssGen = [1;4];
%% Bus data
% bus data format
% bus:
% col1 number
% col2 voltage magnitude(pu)
% col3 voltage angle(degree)
% col4 p_gen(pu)
% col5 q_gen(pu),
% col6 p_load(pu)
% col7 q_load(pu)
% col8 G shunt(pu)
% col9 B shunt(pu)
% col10 bus_type
%     bus_type  - 1, swing bus
%               - 2, generator bus (PV bus)
%               - 3, load bus (PQ bus)
% col11 q_gen_max(pu)
% col12 q_gen_min(pu)
% col13 v_rated (kV)
% col14 v_max  pu
% col15 v_min  pu

PDC=25;

bus = [...
% bus    V        Angle     P_gen     Q_gen     P_load    Q_load         G
B     type QgenMax   QgenMin   V_rate    vmax vmin
   1    1.0250   36.4500      12.0         0         0         0         0
0       2     23       -23        20    1.5  0.5; %Colstrip Gen 14 bus
   2    1.0258   33.1100         0         0    0.0000   -0.0000         0
4       3    0.0       0.0       500    1.5  0.5; %HV bus for Gen 14, North
   3    1.0179   22.2200         0         0   -0.0000    0.0000         0
0       3    0.0       0.0       500    1.5  0.5; %HV Bus 85
   4    1.0318   19.5800         0         0   -0.0000   -0.0000         0
4       3    0.0       0.0       500    1.5  0.5; %HV Bus (Garrison)
   5    1.0058   16.4000         0         0    0.0000    0.0000         0
4       3    0.0       0.0       500    1.5  0.5; %HV bus (HOT SPRINGS)

   6    1.0127   33.7200         0         0   -0.0000   -0.0000         0
25      3    0.0       0.0       500    1.5  0.5; %HV bus (Alberta)
   7    1.0250   37.3700       72.5         0         0         0         0
0       2    130      -130        20    1.5  0.5; %Alberta Gen 34 bus
```

```
   8    1.0000   29.5300         0         0        70      17.5         0
0       3    0.0       0.0       110    1.5  0.5; %Load bus

   9    1.0058    3.3800         0         0    0.0000    0.0000         0
0       3    0.0       0.0       500    1.5  0.5; %HV bus (Ashe)
  10    1.0217   -5.2200         0         0   -0.0000   -0.0000         0
5       3    0.0       0.0       500    1.5  0.5; %HV bus (Schultz)
  11    1.0217   -0.9800         0         0   -0.0000   -0.0000         0
0       3    0.0       0.0       500    1.5  0.5; %HV bus (GrandCoulee)
  12    1.0250       0.0        10    3.1060         0         0         0
0       1     83       -83        20    1.5  0.5; %GrandCoulee Gen 7 (swing
bus)
  13    1.0000     -0.98         0         0         0         0         0
0       3    0.0       0.0       500    1.5  0.5; %HV Bus (ChiefJoseph)

  14    1.0214   -24.950         0         0   -0.0000   -0.0000         0
32.5       3    0.0       0.0       500    1.5  0.5; %HV bus for bus 8 load,
BC
  15    0.9923   -27.870         0         0        45      11.25         0
0       3    0.0       0.0       110    1.5  0.5; %Load bus

  16    1.0214     4.120         0         0   -0.0000   -0.0000         0
10       3    0.0       0.0       500    1.5  0.5; %HV bus for bus 8 load, BC
  17    1.0250    11.370        50    0.3847         0         0         0
0       2     45       -45        20    1.5  0.5; %Gen 1 bus
  18    1.0063     1.740         0         0        35      8.75         0
0       3    0.0       0.0       110    1.5  0.5; %Load bus

  19    1.0224     0.140         0         0       PDC    0.0000         0
0       3    0.0       0.0       500    1.5  0.5; %HV bus (Big Eddy)
  20    1.0000    -5.350         0         0         5      1.25         0
0       3    0.0       0.0       110    1.5  0.5; %Load bus
  21    1.0250     8.750      30.0      2.1979      0         0         0
0       2     36       -36        20    1.5  0.5; %Gen 8 bus
  22    1.0250     5.330      40.0      2.3076      0         0         0
0       2     50       -30        20    1.5  0.5; %Gen 10 bus
  23    1.0140   -10.210         0         0    0.0000    0.0000         0
8       3    0.0       0.0       500    1.5  0.5; %HV bus Captain Jack
  24    1.0140   -10.210         0         0    0.0000    0.0000         0
8       3    0.0       0.0       500    1.5  0.5; %HV bus Malin
  25    1.0188   -18.780         0         0   -0.0000   -0.0000         0
45      3    0.0       0.0       500    1.5  0.5; %HV bus between Malin and
Vincent
  26    1.0250   -12.570       100    4.5989         0         0         0
0       2     52       -52        20    1.5  0.5; %Gen 18 bus
  27    1.0250   -21.090         0         0       100        25         0
0       3    0.0       0.0       110    1.5  0.5; %Load bus
  28    1.0273   -37.560         0         0      -PDC     0.0000         0
100      3    0.0       0.0       500    1.5  0.5; %HV bus Vincent
  29    1.0250   -31.350       170    0.3951         0         0         0
0       2    178      -178        20    1.5  0.5; %Palo Verde Gen 21 bus
  30    1.0026   -46.250         0         0       200        50         0
0       3    0.0       0.0       110    1.5  0.5; %Load bus
  31    1.0234   -41.540         0         0    0.0000    0.0000         0
20      3    0.0       0.0       500    1.5  0.5; %HV bus for Gen 24, 25,
load bus 56, sWest
```

```
  32    1.0250   -46.310        -24   1.6470      0         0            0
0       2   1000     -1000        20    1.5  0.5]; %Gen 24 (Motor) bus

%% Line data
AlbCon2 = 0.01;
if AlbCon; AlbCon2 = 1; end

A = 7/8;

line = [
%         Fbus         Tbus     R          X                   Y
            1            2            0    0.0052174           0; %Gen 1
(Colstrip) stepup x-former
            6            7            0   0.00092308           0; %Gen 2
(Alberta) stepup x-former
           11           12            0    0.0014458           0; %Gen 3
(GrandCoulee) stepup x-former
           16           17            0    0.0026667           0; %Gen 4
(Vancouver) stepup x-former
           21            9            0    0.0033333           0; %Gen 8
stepup x-former
           22           19            0       0.0024           0; %Gen 10
(JohnDay) stepup x-former
           26           25            0    0.0011538           0; %Gen 18
stepup x-former
           29           28            0   0.00067416           0; %Gen 7
(PaloVerde) stepup x-former
           32           31            0      0.00375           0; %Motor
stepup x-former

            6            8            0    0.0010753           0; %Load bus
120 x-former
           14           15            0    0.0016667           0; %Load bus
11 x-former
           16           18            0      0.00125           0; %Load bus
8 x-former
           19           20            0         0.02           0; %Load bus
26 x-former
           27           25            0   0.00043103           0; %Load bus
43 x-former
           30           28            0   0.00076923           0; %Load bus
50 x-former


            2            3       0.0028      0.03504           0; %Colstrip
to Garrison
            3            4       0.0007      0.00876           0; %Garrison
            2            4       0.0035       0.0438           0; %Garrison
            4            5       0.0006       0.0111           0; %Garrison
to HotSprings
            5            6       0.0089       0.1318           0;
%HotSprings to Alberta

            5            9      0.00215      0.03135           0; %Ashe to
HotSprings
```

```
            9           10       0.0011       0.0207           0; %Ashe to
Schultz
            9           19       0.0000       0.002            0; %Ashe to
BigEddy
           10           11       0.0004       0.0063           0; %Schultz
to GrandCoulee
           11           13       0.0004       0.0063           0;
%GrandCoulee to ChiefJoseph
           10           14       0.0013       0.01255          0; %Schultz
to Seattle
           14           16       0.010789     0.03692          0; %Seattle
to Vancouver
           10           19       0.000343     0.007409         0; %Schultz
to BigEddy

           19           23    2*0.001012*A    2*0.008466*A         0;
%BigEddy to CaptainJack
           19           24    2*0.001012*A    2*0.008466*A         0;
%BigEddy to Malin
           23           24      0.00000         0.0001            0;
%CaptainJack to Malin
           23           25   2*0.0009739*A*.8   2*0.007686*A*.8     0;
%Parallel line leaving CaptainJack
           24           25   2*0.0009739*A*.8   2*0.007686*A*.8     0;
%Parallel line leaving Malin
           25           28   2*0.00143*A*1.2    2*0.01578*A*1.2     0;
%Parallel to Vincent
           25           28   2*0.00143*A*1.2    2*0.01578*A*1.2     0;
%Parallel to Vincent
           28           31      0.000322*A      0.00433*A           0;
%Vincent to Mead
            4           31        0.0473*A       0.2062*A
0];%Montana to Mead

% Machine data format
%   Column
%     1. machine number (may be different from bus number),
%     2. bus number,
%     3. machine base mva,
%     4. leakage reactance x_l(pu),
%     5. resistance r_a(pu),
%     6. d-axis sychronous reactance x_d(pu),
%     7. d-axis transient reactance x'_d(pu),
%     8. d-axis subtransient reactance x"_d(pu),
%     9. d-axis open-circuit time constant T'_do(sec),
%    10. d-axis open-circuit subtransient time constant T"_do(sec),
%    11. q-axis sychronous reactance x_q(pu),
%    12. q-axis transient reactance x'_q(pu),
%    13. q-axis subtransient reactance x"_q(pu),
%        NOTE:  PST requires that x"_q = x"_d
%    14. q-axis open-circuit time constant T'_qo(sec),
%        if T'_q0=0, PST sets it to 999 (i.e., removes from model),
%    15. q-axis open circuit subtransient time constant T"_qo(sec),
%        if T"_q0=0, PST sets it to 999,
%    16. inertia constant H(sec),
%    17. damping coefficient d_o(pu),
```

```
%    18. dampling coefficient d_1(pu), (relative to pmech)
%    19. bus number
%    20. s(1.0)
%    21. s(1.2)
% John Undrill reccomended the x'_q, T'_qo, and T"_qo values for the
% salient machines.
d_0S = 0; %Salient damping
d_0R = 0; %Round rotor damping
d_1 = 1; %Pmech damping (Not used in new mac_sub)
mac_con = [
% 1   2   3      4x    5y   6y   7y    8y    9y     10y    11y   12y   13y
14y   15y    16y   17    18   19  20x    21x
% num bus base   x_l   r_a  x_d  x'_d  x"_d  T'_do  T"_do  x_q   x'_q  x"_q
T'_qo T"_qo  H     d_0   d_1  bus s(1.0) s(1.2)
   1   1  2300   0.17  0.0  2.0  0.2   0.18  7.5    0.025  1.9   0.7   0.18
0.5   0.06   6.5   d_0R  d_1   1  0.05   0.3; %Steam, round rotor (Colstip)
   2   7 13000   0.17  0.0  2.0  0.2   0.18  7.5    0.025  1.9   0.7   0.18
0.5   0.06   6.5   d_0R  d_1   7  0.05   0.3; %Steam, round rotor (Alberta)
   3  12  9000   0.17  0.0  1.2  0.3   0.22  6.0    0.025  0.7   0.23  0.22
0.06  0.04   5.0   d_0S  d_1  12  0.05   0.3; %Hydro, salient pole
(GrandCoulee)
   4  17  9000   0.17  0.0  1.2  0.3   0.22  6.0    0.025  0.7   0.23  0.22
0.06  0.04   5.0   d_0S  d_1  17  0.05   0.3; %Hydro, salient pole
(Vancouver)
   5  21  6000   0.17  0.0  1.2  0.3   0.22  6.0    0.025  0.7   0.23  0.22
0.06  0.04   5.0   d_0S  d_1  21  0.05   0.3; %Hydro, salient pole
(UpperColombia)
   6  22  8500   0.17  0.0  1.2  0.3   0.22  6.0    0.025  0.7   0.23  0.22
0.06  0.04   5.0   d_0S  d_1  21  0.05   0.3; %Hydro, salient pole (JohnDay)
   7  26 11000   0.17  0.0  2.0  0.2   0.18  7.5    0.025  1.9   0.7   0.18
0.5   0.06   6.5   d_0R  d_1  26  0.05   0.3; %Gas Turbine, round rotor
   8  29 23900   0.17  0.0  2.0  0.2   0.18  7.5    0.025  1.9   0.7   0.18
0.5   0.06   6.5   d_0R  d_1  29  0.05   0.3; %Steam, round rotor? LA
   9  32 20000   0.17  0.0  2.0  0.2   0.18  7.5    0.025  1.9   0.7   0.18
0.5   0.06   6.5   d_0R  d_1  32  0.05   0.3]; %Steam, round rotor


% exciter model
% exc_con matrix format
%column        data
%  1    exciter type (=3 for exc_st3, IEEE type ST3)
%  2    machine number
%  3    transducer filter time constant (T_R - sec)
%  4    voltage regulator gain (K_A)
%  5    voltage regulator time constant (T_A - sec)
%  6    transient gain reduction time constnat (T_B - sec) -- denominator
%  7    transient gain reduction time constnat (T_C - sec) -- numerator
%  8    max voltage regulator output (V_Rmax - pu)
%  9    min voltage regulator output (V_Rmin - pu)
% 10    max internal signal (VImax - pu)
% 11    min internal signal (VImin - pu)
% 12    first state regulator gain (KJ)
% 13    potential circuit gain coef (KP)
% 14    potential circuit phase angle (qP - degrees)
% 15    current circuit gain coef (KI)
% 16    potential source reactance (XL - pu)
```

```
% 17    rectifier loading factor (KC)
% 18    max field voltage (Efdmax - pu)
% 19    inner loop feedback constant (KG)
% 20    max innerloop voltage feedback (VGmax - pu)
exc_con = [...
%   1   2   3     4     5     6     7    8     9     10    11    12  13  14
15  16  17 18?    19  20
%  type num T_R   K_A   T_A   T_B   T_C  Vrmax Vrmin VImax VImin KJ  KP  qP
KI  XL  KC Efdmax KG  VGmax
    3    1  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3    2  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3    3  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3    4  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3    5  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3    6  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3    7  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3    8  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3    9  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0];

% PSS model
% pss_con matrix format
%column        data         unit
%  1      Type input (1=spd, 2=power)
%  2      machine number
%  3      gain K
%  4      Washout time const Tw (sec)
%  5      1st lead time const T1 (sec)
%  6      1st lag time const T2 (sec)
%  7      2nd lead time const T3 (sec)
%  8      2nd lag time const T4 (sec)
%  9      max output limit (pu)
%  10     min output limit (pu)
pss_con = [];
for k=1:length(pssGen)
%                   type gen#      K  Tw T1   T2   T3   T4    max min
    pss_con(k,:) = [1    pssGen(k) 20 2  0.25 0.04 0.2  0.03  0.1 -0.1];
end

%Gens 1, 2, 13, 14, and 26,

% governor model
% tg_con matrix format
%column        data         unit
%  1    turbine model number (=1)
%  2    machine number
%  3    speed set point   wf        pu
%  4    steady state gain 1/R       pu
```

```
%  5    maximum power order  Tmax   pu on generator base
%  6    servo time constant   Ts    sec
%  7    governor time constant  Tc  sec
%  8    transient gain time constant T3 sec
%  9    HP section time constant   T4   sec
% 10    reheater time constant    T5    sec
tg_con = [...
%     num  wf 1/R        Tmax  Ts    Tc   T3     T4     T5
1     1   1  20         1.0   0.10  10   3.0    0      0.01;%Steam (Colstrip)
1     2   1  20*0.25    1.0   0.10  10   3.0    0      0.01;%Steam (Alberta)
1     3   1  20         1.0   0.50  15   1.0   -1      0.50;%Slow Hydro
(GrandCoulee)
1     4   1  20         1.0   0.50  15   1.0   -1      0.50;%Slow Hydro
(Canada)
1     5   1  20         1.0   0.50  15   1.0   -1      0.50;%Slow Hydro
(UpperColombia)
1     6   1  20         1.0   0.50  15   1.0   -1      0.50;%Slow Hydro
(JohnDay)
1     7   1  20*0.25    1.0   0.50  10   4.0    0      0.01;%Gas Turbine -
75% BASE LOADED
1     8   1  20         1.0   0.10  10   3.0    0      0.01];%Steam
(PaloVerde)


% non-conforming load
% col 1       bus number
% col 2       fraction const active power load
% col 3       fraction const reactive power load
% col 4       fraction const active current load
% col 5       fraction const reactive current load
% load_con = [...
% %bus Pcont Qconst P_Icont Q_Iconst
%   2   0     0     1       0; %Load
%   4   0     0     1       0; %Load
%   6   0     0     1       0; %Load
%   8   0     0     1       0; %Load
%  10   0     0     1       0; %PDCI North
%  12   0     0     1       0; %PDCI South
%  14   0     0     1       0];%Load
%                             %TCSC bus
% load_con=[...
%   13   0     0     1       0];%Load
%
% Load modulation data (sets up modulation of real part of load)
% col 1       load modulation number (Index for b_lmod and lmod_sig)
% col 2       bus number
% col 3       modulation base MVA (MVA)
% col 4       max conductance (pu)
% col 5       min conductance (pu)
% col 6       regulator gain (K)
% col 7       regulator time constant (TR)
% NOTE: This creates b_lmod for the linear analysis.
% lmod_con=[...
% %num bus  MVA  Max  Min  K  TR
%   1   2   100  100 -100  1  0.05; %BC
%   2   4   100  100 -100  1  0.05; %BC
```

```
%   3   6   100  100 -100  1  0.05; %BC
%   4   8   100  100 -100  1  0.05; %BC
%   5  10   100  100 -100  1  0.05; %BC
%   6  12   100  100 -100  1  0.05; %BC
%   7  14   100  100 -100  1  0.05; %North
%   8  16   100  100 -100  1  0.05; %North
%   9  18   100  100 -100  1  0.05; %North
%  10  20   100  100 -100  1  0.05];%North
%                                   %Load
% lmod_con=[...
%    1  13   100  100 -100  1  0.05]; %North
%
% Reactive load modulation matrix.
% col 1       load modulation number (Index for b_rlmod and rlmod_sig)
% col 2       bus number
% col 3       modulation base MVA (MVA)
% col 4       max conductance (pu)
% col 5       min conductance (pu)
% col 6       regulator gain (K)
% col 7       regulator time constant (TR)
% NOTE: This creates b_rlmod for the linear analysis.
% rlmod_con=[...
% %num bus  MVA  Max  Min  K  TR
%   1   2   100  100 -100  1  0.05; %BC
%   2   4   100  100 -100  1  0.05; %BC
%   3   6   100  100 -100  1  0.05; %BC
%   4   8   100  100 -100  1  0.05; %BC
%   5  10   100  100 -100  1  0.05; %BC
%   6  12   100  100 -100  1  0.05; %BC
%   7  14   100  100 -100  1  0.05; %North
%   8  16   100  100 -100  1  0.05; %North
%   9  18   100  100 -100  1  0.05; %North
%  10  20   100  100 -100  1  0.05]; %North
%                                   %Load

% Monitored lines.  When conducting an eigenanalysis, this causes
% c_pf1, c_pf2, c_qf1, c_qf2, c_ilif, c_ilit, c_ilmf, c_ilmt, c_ilrf,
% c_ilrt to be created.  The ith row of c_pf1
% and c_qf1 correspond to line(lmon_con(i),:) of the line matrix.
lmon_con = [1:size(line,1)]; %All lines
```

24. Appendix O: MicroWECC V2 PST Data File

```
% MicroWECC V2
% Based on d_minniWECC_V3C_C3_6.m
% PST data file
%
%
% System bases:  100 MVA, 60 Hz.

% Written by RJ Hallett, 2018

pssGen = [1;4];

%% Operating parameters
% PDC = DC line real-power flows
PDC = [28.5];

%% Bus data
% bus data format
% bus:
% col1 number
% col2 voltage magnitude(pu)
% col3 voltage angle(degree)
% col4 p_gen(pu)
% col5 q_gen(pu),
% col6 p_load(pu)
% col7 q_load(pu)
% col8 G shunt(pu)
% col9 B shunt(pu)
% col10 bus_type
%     bus_type  - 1, swing bus
%               - 2, generator bus (PV bus)
%               - 3, load bus (PQ bus)
% col11 q_gen_max(pu)
% col12 q_gen_min(pu)
% col13 v_rated (kV)
% col14 v_max  pu
% col15 v_min  pu
bus = [...
% bus    V        Angle     P_gen     Q_gen     P_load    Q_load         G
B     type QgenMax   QgenMin   V_rate    vmax vmin
   1    1.0250   36.4500     14.295        0         0         0         0
0       2     23       -23        20    1.5  0.5; %Colstrip Gen 14 bus
   2    1.0258   33.1100         0         0    0.0000   -0.0000         0
0       3    0.0       0.0       500    1.5  0.5; %HV bus for Gen 14, North
   3    1.0179   22.2200         0         0   -0.0000    0.0000         0
0       3    0.0       0.0       500    1.5  0.5; %HV Bus 85
   4    1.0318   19.5800         0         0   -0.0000   -0.0000         0
5       3    0.0       0.0       500    1.5  0.5; %HV Bus (Garrison)
   5    1.0058   16.4000         0         0    0.0000    0.0000         0
5       3    0.0       0.0       500    1.5  0.5; %HV bus (Taft)
   6    1.0058    2.6363         0         0    0.0000    0.0000         0
0.0000       3    0.0       0.0       500    1.5  0.5; %HV bus for Gen 12 and
bus 29 load, North
```

```
   7    1.0167   26.0405         0         0    0.0000   -0.0000         0
0.0000       3    0.0       0.0       500    1.5  0.5; %HV bus for Gen 3, BC
   8    1.0127   33.7200         0         0   -0.0000   -0.0000         0
28.25        3    0.0       0.0       500    1.5  0.5; %HV bus (Alberta)
   9    1.0250   37.3700      84.5         0         0         0         0
0       2    130      -130        20    1.5  0.5; %Alberta Gen 34 bus
  10    1.0000   29.5300         0         0        82      20.5         0
0       3    0.0       0.0       110    1.5  0.5; %Load bus

  11    1.0058    3.3800         0         0    0.0000    0.0000         0
0       3    0.0       0.0       500    1.5  0.5; %HV bus (Ashe)
  12    1.0217   -5.2200         0         0   -0.0000   -0.0000         0
4       3    0.0       0.0       500    1.5  0.5; %HV bus (Schultz)
  13    1.0217   -0.9800         0         0   -0.0000   -0.0000         0
0       3    0.0       0.0       500    1.5  0.5; %HV bus (GrandCoulee)
  14    1.0250       0.0        10    3.1060         0         0         0
0       1     83       -83        20    1.5  0.5; %GrandCoulee Gen 7 (swing
bus)
  15    1.0000     -0.98         0         0         0         0         0
0       3    0.0       0.0       500    1.5  0.5; %HV Bus (ChiefJoseph)

  16    1.0214   -24.950         0         0   -0.0000   -0.0000         0
31       3    0.0       0.0       500    1.5  0.5; %HV bus for bus 8 load, BC
  17    0.9923   -27.870         0         0        45      11.25         0
0       3    0.0       0.0       110    1.5  0.5; %Load bus

  18    1.0214     4.120         0         0   -0.0000   -0.0000         0
10       3    0.0       0.0       500    1.5  0.5; %HV bus for bus 8 load, BC
  19    1.0250    11.370        48    0.3847         0         0         0
0       2     45       -45        20    1.5  0.5; %Gen 1 bus
  20    1.0063     1.740         0         0        42      10.5         0
0       3    0.0       0.0       110    1.5  0.5; %Load bus

  21    1.0224     0.140         0         0      PDC     0.0000         0
40       3    0.0       0.0       500    1.5  0.5; %HV bus (Big Eddy)
  22    1.0000    -5.350         0         0     13.5       3.75         0
0       3    0.0       0.0       110    1.5  0.5; %Load bus
  23    1.0250     8.750      40.0      2.1979      0         0         0
0       2     36       -36        20    1.5  0.5; %Gen 8 bus
  24    1.0250     5.330      58.0      2.3076      0         0         0
0       2     50       -30        20    1.5  0.5; %Gen 10 bus
  25    1.0140   -10.210         0         0    0.0000    0.0000         0
4       3    0.0       0.0       500    1.5  0.5; %HV bus Captain Jack
  26    1.0140   -10.210         0         0    0.0000    0.0000         0
4       3    0.0       0.0       500    1.5  0.5; %HV bus Malin
  27    1.0188   -18.780         0         0   -0.0000   -0.0000         0
45      3    0.0       0.0       500    1.5  0.5; %HV bus between Malin and
Vincent
  28    1.0250   -12.570        95    4.5989         0         0         0
0       2     52       -52        20    1.5  0.5; %Gen 18 bus
  29    1.0250   -21.090         0         0       095     23.75         0
0       3    0.0       0.0       110    1.5  0.5; %Load bus
  30    1.0273   -37.560         0         0      -PDC     0.0000         0
105      3    0.0       0.0       500    1.5  0.5; %HV bus Vincent
  31    1.0250   -31.350       170    0.3951         0         0         0
0       2    178      -178        20    1.5  0.5; %Palo Verde Gen 21 bus
```

```
  32    1.0026   -46.250         0         0       220        50         0
0       3    0.0       0.0       110    1.5  0.5; %Load bus
  33    1.0234   -41.540         0         0    0.0000    0.0000         0
70      3    0.0       0.0       500    1.5  0.5; %HV bus for Gen 24, 25,
load bus 56, sWest
  34    1.0250   -46.310       100    1.6470         0         0         0
0       2    100      -100        20    1.5  0.5; %Gen 24 (Motor) bus
  35    1.0086  -40.6270         0         0       108        27         0
0       3    0.0       0.0       110    1.5  0.5]; %Load bus

%% Line data
line = [
%         Fbus          Tbus     R          X                    Y
            1             2            0    0.0052174            0; %Gen 1
(Colstrip) stepup x-former
            8             9            0   0.00092308            0; %Gen 2
(Alberta) stepup x-former
            13            14           0    0.0014458            0; %Gen 3
(GrandCoulee) stepup x-former
            18            19           0    0.0026667            0; %Gen 4
(BC) stepup x-former
            23            11           0    0.0033333            0; %Gen 5
stepup x-former
            24            21           0       0.0024            0; %Gen 6
(JohnDay) stepup x-former
            28            27           0    0.0011538            0; %Gen 7
stepup x-former
            31            30           0   0.00067416            0; %Gen 8
(PaloVerde) stepup x-former
            34            33           0    0.00375           0; %Gen 9
stepup x-former

            8             10           0    0.0010753            0; %Load bus
120 x-former
            16            17           0    0.0016667            0; %Load bus
11 x-former
            18            20           0      0.00125            0; %Load bus
8 x-former
            21            22           0         0.02            0; %Load bus
26 x-former
            29            27           0   0.00043103            0; %Load bus
43 x-former
            32            30           0   0.00076923            0; %Load bus
50 x-former
            35            33           0      0.00125            0; %Load bus
56 x-former

             2             3       0.0028      0.03504            0; %L1
Colstrip to Garrison
             2             4       0.0035       0.0438            0; %L2
Colstrip to Garrison
             3             4       0.0007      0.00876            0; %Garrison
             4             5       0.0006       0.0111            0; %Garrison
to Taft
             5             6       0.0012      0.0202            0; %Taft to
Bell
```

```
            6            7       0.003125     0.0375           0; %Bell to
Selkirk
            7            8       0.00460      0.0741           0; %Selkirk
to Alberta

            7           18       0.002761     0.0285           0; %Selkirk
to B.C.
            6           13       0.00064      0.01456          0; %Bell to
G.C.
            5           11       0.00215      0.03135          0; %Ashe to
Taft
           11           12       0.0011       0.0207           0; %Ashe to
Schultz
           11           21       0.0000       0.002            0; %Ashe to
BigEddy
           12           13       0.0004       0.0063           0; %Schultz
to GrandCoulee
           13           15       0.0004       0.0063           0;
%GrandCoulee to ChiefJoseph
           12           16       0.0013       0.01255          0; %Schultz
to Seattle
           16           18       0.010789     0.03692          0; %Seattle
to BC
           12           21       0.000343     0.007409         0; %Schultz
to BigEddy

           21           25    2*0.0007721  2*0.0071427869          0;
%BigEddy to CaptainJack
           21           26    2*0.0007721  2*0.0071427869       0; %BigEddy
to Malin
           25           26       0.00000      0.0001              0;
%CaptainJack to Malin
           25           27       2*0.0009737   2*0.007686         0;
%CaptainJack to N. Cali
           26           27       2*0.0009737   2*0.007686         0; %Malin
to N. Cali
           27           30       2*0.0013139   2*0.015884         0;
%Parallel to Vincent
           27           30       2*0.0013139   2*0.015884         0;
%Parallel to Vincent
           30           33      0.000322      0.00433          0; %Vincent
to Mead
            4           33        0.0473       0.2062           0];%Montana
to Mead

% Machine data format
%   Column
%     1. machine number (may be different from bus number),
%     2. bus number,
%     3. machine base mva,
%     4. leakage reactance x_l(pu),
%     5. resistance r_a(pu),
%     6. d-axis sychronous reactance x_d(pu),
%     7. d-axis transient reactance x'_d(pu),
%     8. d-axis subtransient reactance x"_d(pu),
%     9. d-axis open-circuit time constant T'_do(sec),
```

```
%    10. d-axis open-circuit subtransient time constant T"_do(sec),
%    11. q-axis sychronous reactance x_q(pu),
%    12. q-axis transient reactance x'_q(pu),
%    13. q-axis subtransient reactance x"_q(pu),
%        NOTE:  PST requires that x"_q = x"_d
%    14. q-axis open-circuit time constant T'_qo(sec),
%        if T'_q0=0, PST sets it to 999 (i.e., removes from model),
%    15. q-axis open circuit subtransient time constant T"_qo(sec),
%        if T"_q0=0, PST sets it to 999,
%    16. inertia constant H(sec),
%    17. damping coefficient d_o(pu),
%    18. dampling coefficient d_1(pu), (relative to pmech)
%    19. bus number
%    20. s(1.0)
%    21. s(1.2)
% John Undrill reccomended the x'_q, T'_qo, and T"_qo values for the
% salient machines.
d_0S = 0; %Salient damping
d_0R = 0; %Round rotor damping
d_1 = 1; %Pmech damping (Not used in new mac_sub)
mac_con = [
% 1   2   3      4x    5y   6y   7y    8y    9y     10y    11y   12y   13y
14y   15y    16y   17    18   19  20x    21x
% num bus base   x_l   r_a  x_d  x'_d  x"_d  T'_do  T"_do  x_q   x'_q  x"_q
T'_qo T"_qo  H     d_0   d_1  bus s(1.0) s(1.2)
   1   1  2300   0.17  0.0  2.0  0.2   0.18  7.5    0.025  1.9   0.7   0.18
0.5   0.06   6.5   d_0R  d_1   1  0.05   0.3; %Steam, round rotor (Colstip)
   2   9 13000   0.17  0.0  2.0  0.2   0.18  7.5    0.025  1.9   0.7   0.18
0.5   0.06   6.5   d_0R  d_1   7  0.05   0.3; %Steam, round rotor (Alberta)
   3  14  8300   0.17  0.0  1.2  0.3   0.22  6.0    0.025  0.7   0.23  0.22
0.06  0.04   5.0   d_0S  d_1  14  0.05   0.3; %Hydro, salient pole
(GrandCoulee)
   4  19 10800   0.17  0.0  1.2  0.3   0.22  6.0    0.025  0.7   0.23  0.22
0.06  0.04   5.0   d_0S  d_1  19  0.05   0.3; %Hydro, salient pole
(Vancouver)
   5  23  5200   0.17  0.0  1.2  0.3   0.22  6.0    0.025  0.7   0.23  0.22
0.06  0.04   5.0   d_0S  d_1  23  0.05   0.3; %Hydro, salient pole
(UpperColombia)
   6  24  8900   0.17  0.0  1.2  0.3   0.22  6.0    0.025  0.7   0.23  0.22
0.06  0.04   5.0   d_0S  d_1  24  0.05   0.3; %Hydro, salient pole (JohnDay)
   7  28 10000   0.17  0.0  2.0  0.2   0.18  7.5    0.025  1.9   0.7   0.18
0.5   0.06   6.5   d_0R  d_1  27  0.05   0.3; %Gas Turbine, round rotor
   8  31 29900   0.17  0.0  2.0  0.2   0.18  7.5    0.025  1.9   0.7   0.18
0.5   0.06   6.5   d_0R  d_1  31  0.05   0.3; %Steam, round rotor? LA
   9  34 28000   0.17  0.0  2.0  0.2   0.18  7.5    0.025  1.9   0.7   0.18
0.5   0.06   6.5   d_0R  d_1  34  0.05   0.3]; %Steam, round rotor


% exciter model
% exc_con matrix format
%column        data
%  1    exciter type (=3 for exc_st3, IEEE type ST3)
%  2    machine number
%  3    transducer filter time constant (T_R - sec)
%  4    voltage regulator gain (K_A)
%  5    voltage regulator time constant (T_A - sec)
```

```
%  6    transient gain reduction time constnat (T_B - sec) -- denominator
%  7    transient gain reduction time constnat (T_C - sec) -- numerator
%  8    max voltage regulator output (V_Rmax - pu)
%  9    min voltage regulator output (V_Rmin - pu)
% 10    max internal signal (VImax - pu)
% 11    min internal signal (VImin - pu)
% 12    first state regulator gain (KJ)
% 13    potential circuit gain coef (KP)
% 14    potential circuit phase angle (qP - degrees)
% 15    current circuit gain coef (KI)
% 16    potential source reactance (XL - pu)
% 17    rectifier loading factor (KC)
% 18    max field voltage (Efdmax - pu)
% 19    inner loop feedback constant (KG)
% 20    max innerloop voltage feedback (VGmax - pu)
exc_con = [...
%   1   2   3     4     5     6     7    8     9     10    11    12  13  14
15  16  17 18?    19  20
%  type num T_R   K_A   T_A   T_B   T_C  Vrmax Vrmin VImax VImin KJ  KP  qP
KI  XL  KC Efdmax KG  VGmax
    3    1  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3    2  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3    3  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3    4  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3    5  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3    6  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3    7  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3    8  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0;
    3    9  0.0   200   0.02  10.0  1.0  5.0   -5.0  0.1   -0.1  1.0 1   0
0   0   0  5.0    0  5.0];

% PSS model
% pss_con matrix format
%column        data         unit
%  1      Type input (1=spd, 2=power)
%  2      machine number
%  3      gain K
%  4      Washout time const Tw (sec)
%  5      1st lead time const T1 (sec)
%  6      1st lag time const T2 (sec)
%  7      2nd lead time const T3 (sec)
%  8      2nd lag time const T4 (sec)
%  9      max output limit (pu)
%  10     min output limit (pu)
pss_con = [];
for k=1:length(pssGen)
%                   type gen#      K  Tw T1   T2   T3   T4    max min
    pss_con(k,:) = [1    pssGen(k) 20 2  0.25 0.04 0.2  0.03  0.1 -0.1];
end
```

```
% governor model
% tg_con matrix format
%column        data         unit
%  1    turbine model number (=1)
%  2    machine number
%  3    speed set point   wf        pu
%  4    steady state gain 1/R       pu
%  5    maximum power order  Tmax   pu on generator base
%  6    servo time constant   Ts    sec
%  7    governor time constant  Tc  sec
%  8    transient gain time constant T3 sec
%  9    HP section time constant   T4   sec
% 10    reheater time constant    T5    sec
tg_con = [...
%     num  wf 1/R        Tmax  Ts    Tc   T3     T4     T5
1     1   1  20         1.0   0.10  10   3.0    0      0.01;%Steam (Colstrip)
1     2   1  20*0.25    1.0   0.10  10   3.0    0      0.01;%Steam (Alberta)
1     3   1  20         1.0   0.50  15   1.0   -1      0.50;%Slow Hydro
(GrandCoulee)
1     4   1  20         1.0   0.50  15   1.0   -1      0.50;%Slow Hydro
(Canada)
1     5   1  20         1.0   0.50  15   1.0   -1      0.50;%Slow Hydro
(UpperColombia)
1     6   1  20         1.0   0.50  15   1.0   -1      0.50;%Slow Hydro
(JohnDay)
1     7   1  20*0.25    1.0   0.50  10   4.0    0      0.01;%Gas Turbine -
75% BASE LOADED
1     8   1  20         1.0   0.10  10   3.0    0      0.01;%Steam
(PaloVerde)
1     9   1  20*0.20    1.0   0.10  10   3.0    0      0.01];%Steam

% non-conforming load
% col 1       bus number
% col 2       fraction const active power load
% col 3       fraction const reactive power load
% col 4       fraction const active current load
% col 5       fraction const reactive current load
% load_con = [...
% %bus Pcont Qconst P_Icont Q_Iconst
%   2   0     0     1       0; %Load
%   4   0     0     1       0; %Load
%   6   0     0     1       0; %Load
%   8   0     0     1       0; %Load
%  10   0     0     1       0; %PDCI North
%  12   0     0     1       0; %PDCI South
%  14   0     0     1       0];%Load
%                             %TCSC bus
% load_con=[...
%   13   0     0     1       0];%Load
%
% Load modulation data (sets up modulation of real part of load)
% col 1       load modulation number (Index for b_lmod and lmod_sig)
% col 2       bus number
% col 3       modulation base MVA (MVA)
% col 4       max conductance (pu)
```

```
% col 5       min conductance (pu)
% col 6       regulator gain (K)
% col 7       regulator time constant (TR)
% NOTE: This creates b_lmod for the linear analysis.
% lmod_con=[...
% %num bus  MVA  Max  Min  K  TR
%   1   2   100  100 -100  1  0.05; %BC
%   2   4   100  100 -100  1  0.05; %BC
%   3   6   100  100 -100  1  0.05; %BC
%   4   8   100  100 -100  1  0.05; %BC
%   5  10   100  100 -100  1  0.05; %BC
%   6  12   100  100 -100  1  0.05; %BC
%   7  14   100  100 -100  1  0.05; %North
%   8  16   100  100 -100  1  0.05; %North
%   9  18   100  100 -100  1  0.05; %North
%  10  20   100  100 -100  1  0.05];%North
%                                   %Load
% lmod_con=[...
%    1  13   100  100 -100  1  0.05]; %North
%
% Reactive load modulation matrix.
% col 1       load modulation number (Index for b_rlmod and rlmod_sig)
% col 2       bus number
% col 3       modulation base MVA (MVA)
% col 4       max conductance (pu)
% col 5       min conductance (pu)
% col 6       regulator gain (K)
% col 7       regulator time constant (TR)
% NOTE: This creates b_rlmod for the linear analysis.
% rlmod_con=[...
% %num bus  MVA  Max  Min  K  TR
%   1   2   100  100 -100  1  0.05; %BC
%   2   4   100  100 -100  1  0.05; %BC
%   3   6   100  100 -100  1  0.05; %BC
%   4   8   100  100 -100  1  0.05; %BC
%   5  10   100  100 -100  1  0.05; %BC
%   6  12   100  100 -100  1  0.05; %BC
%   7  14   100  100 -100  1  0.05; %North
%   8  16   100  100 -100  1  0.05; %North
%   9  18   100  100 -100  1  0.05; %North
%  10  20   100  100 -100  1  0.05]; %North
%                                   %Load

% Monitored lines.  When conducting an eigenanalysis, this causes
% c_pf1, c_pf2, c_qf1, c_qf2, c_ilif, c_ilit, c_ilmf, c_ilmt, c_ilrf,
% c_ilrt to be created.  The ith row of c_pf1
% and c_qf1 correspond to line(lmon_con(i),:) of the line matrix.
lmon_con = [1:size(line,1)]; %All lines
```

25. Appendix P: Spring-Mass Case 1

```
clear variables; clc; close all;
% Script that:
% 1. Builds a 3-mass, 2 spring system in SS form
% 2. Excites first mass with velocity impulse of magnitude 1
%    and plots speed response of system
% 3. Performs modal analysis
%
%Author: RJ Hallett
%Date 06/18
%% Define constants
% Spring 1 and 2 coeffecients
k1 = 1;
k2 = 10;
% Ground friction coefficient
do = 0.00;
% Damper 1 and 2 coefficients
d1 = 0.00;
d2 = 0.00;
% Mass values
m1 = 1;
m2 = .5;
m3 = 20;
%% Build State-Space Matrices
A = [ 0 1 0 0 0 0;
      -k1/m1 -(do+d1)/m1 k1/m1 d1/m1 0 0;
      0 0 0 1 0 0;
      k1/m2 d1/m2 -(k1+k2)/m2 -(do+d1+d2)/m2 k2/m2 d2/m2;
      0 0 0 0 0 1;
      0 0 k2/m3 d2/m3 -k2/m3 -(do+d2)/m3];
B = [ 0; 1/m1; 0; 0; 0; 0];
C = eye(6);
G = ss(A,B,C,zeros(6,1));   %Create ss model
%% Simulate SS model
t = 0:.001:30;          %Time vector
f = zeros(length(t),1); %Initialize input vector
f(1) = 1000;            %Input of size 1/timestep creates velocity magnitude
of 1
                        %at first timestep
%Perform linear simulation; system G, input U, time vector t
[y,t]=lsim(G,f,t);
%Plot velocity of each mass
figure
plot(t,y(:,2),'k','linewidth', 3)
hold on
plot(t,y(:,4),'Color',[0,0.3,0.8],'linewidth', 3)
plot(t,y(:,6),'Color',[0,0.7,0.2],'linewidth', 3)
title('Case 1 Speed Response')
xlabel('Time (s)')
ylabel('Speed')
axis([0,t(end),-1,1.5])
legend('Mass 1','Mass 2','Mass 3','location','northeast')
grid on
ax = gca;
ax.GridAlpha = 1;
```

```
set(gca,'gridlinestyle','-');

figure
plot(t,y(:,1),'k','linewidth', 3)
hold on
plot(t,y(:,3),'Color',[0,0.3,0.8],'linewidth', 3)
plot(t,y(:,5),'Color',[0,0.7,0.2],'linewidth', 3)
xlabel('Time (s)')
ylabel('Displacement')
title('Case 1 Displacement Response')
axis([0,t(end),-1,3])
legend('Mass 1','Mass 2','Mass 3','location','northeast')
grid on
ax = gca;
ax.GridAlpha = 1;
set(gca,'gridlinestyle','-');
%% Modal Analysis
% Find right eigenvalues u, and diagonal matrix of eigenvalues D
[U,D] =eig(A);
l = diag(D);    %Store eigenvalues in l

% List frequency of modes in Hz and Damping percent
Modes = [[1:length(l)]' imag(l)./(2*pi) -100*real(l)./abs(l)]

% Calculate mode shapes
for EigNum = 1:length(l)    %For each eigenvalue
    Index = [2,4,6];        %Index for velocity states
    Modeshape=(180/pi)*angle(U(Index, EigNum));  %Mode shape angle in deg
    ModeAmp = abs(U(Index, EigNum))./max(abs(U(Index, EigNum))); %Mode
amplitudes
    x=[[1:3]' ModeAmp Modeshape];   %Store amplitude and angle
    [~,i]=sort(x(:,2),'descend'); %index of amplitudes in descending order
    x=x(i,:);                     %Reorder matrix using index
    x(:,3)=x(:,3)-x(1,3)          %Normalize angle to 0 and display

    %% Plot Mode Shapes
    figure

    %Create plot limits
    max_lim = 1;
    x_fake=[0 max_lim 0 -max_lim];
    y_fake=[max_lim 0 -max_lim 0];
    h_fake=compass(x_fake,y_fake);
    hold on;
    set(h_fake,'Visible','off','HandleVisibility','off')
    % Plot amplitude and angle of modes
    [~,maxI] = max(abs(U(Index, EigNum)));
    h = compass(U(Index(1),EigNum)./U(Index(maxI),EigNum),'k');
    set(h,'LineWidth',3)
    h = compass(U(Index(2),EigNum)./U(Index(maxI),EigNum));
    set(h,'LineWidth',3,'Color',[0,0.3,0.8])
    h = compass(U(Index(3),EigNum)./U(Index(maxI),EigNum));
    set(h,'LineWidth',3,'Color',[0,0.7,0.2])
    title(['Shape of ', num2str(imag(l(EigNum))/(2*pi)), ' Hz Mode'])
    legend('Mass 1','Mass 2', 'Mass 3','location','southoutside')
end
```

26. Appendix Q: Spring-Mass Case 2

```
clear variables; clc; close all;
% Script that:
% 1. Builds a 3-mass, 2 spring system in SS form
% 2. Excites first mass with velocity impulse of magnitude 1
%    and plots speed response of system
% 3. Performs modal analysis
%
%Author: RJ Hallett
%Date 06/18
%% Define constants
% Spring 1 and 2 coeffecients
k1 = 1;
k2 = 0.2;
% Ground friction coefficient
do = 0.0;
% Damper 1 and 2 coefficients
d1 = 0.00;
d2 = 0.00;
% Mass values
m1 = 1;
m2 = 5;
m3 = 2;
%% Build State-Space Matrices
A = [ 0 1 0 0 0 0;
      -k1/m1 -(do+d1)/m1 k1/m1 d1/m1 0 0;
      0 0 0 1 0 0;
      k1/m2 d1/m2 -(k1+k2)/m2 -(do+d1+d2)/m2 k2/m2 d2/m2;
      0 0 0 0 0 1;
      0 0 k2/m3 d2/m3 -k2/m3 -(do+d2)/m3];
B = [ 0; 1/m1; 0; 0; 0; 0];
C = eye(6);
G = ss(A,B,C,zeros(6,1));   %Create ss model
%% Simulate SS model
t = 0:.001:30;          %Time vector
f = zeros(length(t),1); %Initialize input vector
f(1) = 1000;            %Input of size 1/timestep creates velocity magnitude
of 1
                        %at first timestep
%Perform linear simulation; system G, input U, time vector t
[y,t]=lsim(G,f,t);
%Plot velocity of each mass
figure
plot(t,y(:,2),'k','linewidth', 3)
hold on
plot(t,y(:,4),'Color',[0,0.3,0.8],'linewidth', 3)
plot(t,y(:,6),'Color',[0,0.7,0.2],'linewidth', 3)
title('Case 2 Speed Response')
xlabel('Time (s)')
ylabel('Speed')
axis([0,t(end),-1,1.5])
legend('Mass 1','Mass 2','Mass 3','location','northeast')
grid on
ax = gca;
ax.GridAlpha = 1;
```

```
set(gca,'gridlinestyle','-');

figure
plot(t,y(:,1),'k','linewidth', 3)
hold on
plot(t,y(:,3),'Color',[0,0.3,0.8],'linewidth', 3)
plot(t,y(:,5),'Color',[0,0.7,0.2],'linewidth', 3)
xlabel('Time (s)')
ylabel('Displacement')
title('Case 2 Displacement Response')
axis([0,t(end),-0.5,5.5])
legend('Mass 1','Mass 2','Mass 3','location','northeast')
grid on
ax = gca;
ax.GridAlpha = 1;
set(gca,'gridlinestyle','-');
%% Modal Analysis
% Find right eigenvalues u, and diagonal matrix of eigenvalues D
[U,D] =eig(A);
l = diag(D);    %Store eigenvalues in l

% List frequency of modes in Hz and Damping percent
Modes = [[1:length(l)]' imag(l)./(2*pi) -100*real(l)./abs(l)]

% Calculate mode shapes
for EigNum = 1:length(l)    %For each eigenvalue
    Index = [2,4,6];        %Index for velocity states
    Modeshape=(180/pi)*angle(U(Index, EigNum));  %Mode shape angle in deg
    ModeAmp = abs(U(Index, EigNum))./max(abs(U(Index, EigNum))); %Mode
amplitudes
    x=[[1:3]' ModeAmp Modeshape];   %Store amplitude and angle
    [~,i]=sort(x(:,2),'descend'); %index of amplitudes in descending order
    x=x(i,:);                     %Reorder matrix using index
    x(:,3)=x(:,3)-x(1,3)          %Normalize angle to 0 and display

    %% Plot Mode Shapes
    figure

    %Create plot limits
    max_lim = 1;
    x_fake=[0 max_lim 0 -max_lim];
    y_fake=[max_lim 0 -max_lim 0];
    h_fake=compass(x_fake,y_fake);
    hold on;
    set(h_fake,'Visible','off','HandleVisibility','off')
    % Plot amplitude and angle of modes
    [~,maxI] = max(abs(U(Index, EigNum)));
    h = compass(U(Index(1),EigNum)./U(Index(maxI),EigNum),'k');
    set(h,'LineWidth',3)
    h = compass(U(Index(2),EigNum)./U(Index(maxI),EigNum));
    set(h,'LineWidth',3,'Color',[0,0.3,0.8])
    h = compass(U(Index(3),EigNum)./U(Index(maxI),EigNum));
    set(h,'LineWidth',3,'Color',[0,0.7,0.2])
    title(['Shape of ', num2str(imag(l(EigNum))/(2*pi)), ' Hz Mode'])
    legend('Mass 1','Mass 2', 'Mass 3','location','southoutside')
end
```

27. Appendix R: Spring-Mass Case 3

```
clear variables; clc; close all;
% Script that:
% 1. Builds a 3-mass, 2 spring system in SS form
% 2. Excites first mass with velocity impulse of magnitude 1
%    and plots speed response of system
% 3. Performs modal analysis
%
%Author: RJ Hallett
%Date 06/18
%% Define constants
% Spring 1 and 2 coeffecients
k1 = 1;
k2 = 0.2;
% Ground friction coefficient
do = 0.1;
% Damper 1 and 2 coefficients
d1 = 0.30;
d2 = 0.010;
% Mass values
m1 = 1;
m2 = 5;
m3 = 2;
%% Build State-Space Matrices
A = [ 0 1 0 0 0 0;
      -k1/m1 -(do+d1)/m1 k1/m1 d1/m1 0 0;
      0 0 0 1 0 0;
      k1/m2 d1/m2 -(k1+k2)/m2 -(do+d1+d2)/m2 k2/m2 d2/m2;
      0 0 0 0 0 1;
      0 0 k2/m3 d2/m3 -k2/m3 -(do+d2)/m3];
B = [ 0; 1/m1; 0; 0; 0; 0];
C = eye(6);
G = ss(A,B,C,zeros(6,1));   %Create ss model
%% Simulate SS model
t = 0:.001:30;          %Time vector
f = zeros(length(t),1); %Initialize input vector
f(1) = 1000;            %Input of size 1/timestep creates velocity magnitude
of 1
                        %at first timestep
%Perform linear simulation; system G, input U, time vector t
[y,t]=lsim(G,f,t);
%Plot velocity of each mass
figure
plot(t,y(:,2),'k','linewidth', 3)
hold on
plot(t,y(:,4),'Color',[0,0.3,0.8],'linewidth', 3)
plot(t,y(:,6),'Color',[0,0.7,0.2],'linewidth', 3)
title('Case 3 Speed Response')
xlabel('Time (s)')
ylabel('Speed')
axis([0,t(end),-.5,1.1])
legend('Mass 1','Mass 2','Mass 3','location','northeast')
grid on
ax = gca;
ax.GridAlpha = 1;
```

```
set(gca,'gridlinestyle','-');

figure
plot(t,y(:,1),'k','linewidth', 3)
hold on
plot(t,y(:,3),'Color',[0,0.3,0.8],'linewidth', 3)
plot(t,y(:,5),'Color',[0,0.7,0.2],'linewidth', 3)
xlabel('Time (s)')
ylabel('Displacement')
title('Case 3 Displacement Response')
axis([0,t(end),-0.5,3.5])
legend('Mass 1','Mass 2','Mass 3','location','northeast')
grid on
ax = gca;
ax.GridAlpha = 1;
set(gca,'gridlinestyle','-');
%% Modal Analysis
% Find right eigenvalues u, and diagonal matrix of eigenvalues D
[U,D] =eig(A);
l = diag(D);    %Store eigenvalues in l

% List frequency of modes in Hz and Damping percent
Modes = [[1:length(l)]' imag(l)./(2*pi) -100*real(l)./abs(l)]

% Calculate mode shapes
for EigNum = 1:length(l)    %For each eigenvalue
    Index = [2,4,6];        %Index for velocity states
    Modeshape=(180/pi)*angle(U(Index, EigNum));  %Mode shape angle in deg
    ModeAmp = abs(U(Index, EigNum))./max(abs(U(Index, EigNum))); %Mode
amplitudes
    x=[[1:3]' ModeAmp Modeshape];   %Store amplitude and angle
    [~,i]=sort(x(:,2),'descend'); %index of amplitudes in descending order
    x=x(i,:);                     %Reorder matrix using index
    x(:,3)=x(:,3)-x(1,3)          %Normalize angle to 0 and display

    %% Plot Mode Shapes
    figure

    %Create plot limits
    max_lim = 1;
    x_fake=[0 max_lim 0 -max_lim];
    y_fake=[max_lim 0 -max_lim 0];
    h_fake=compass(x_fake,y_fake);
    hold on;
    set(h_fake,'Visible','off','HandleVisibility','off')
    % Plot amplitude and angle of modes
    [~,maxI] = max(abs(U(Index, EigNum)));
    h = compass(U(Index(1),EigNum)./U(Index(maxI),EigNum),'k');
    set(h,'LineWidth',3)
    h = compass(U(Index(2),EigNum)./U(Index(maxI),EigNum));
    set(h,'LineWidth',3,'Color',[0,0.3,0.8])
    h = compass(U(Index(3),EigNum)./U(Index(maxI),EigNum));
    set(h,'LineWidth',3,'Color',[0,0.7,0.2])
    title(['Shape of ', num2str(imag(l(EigNum))/(2*pi)), ' Hz Mode'])
    legend('Mass 1','Mass 2', 'Mass 3','location','southoutside')
end
```

28. Appendix S: Generate Linear MicroWECC Model Code

```
%% Build A, B, C, D matrices (linear model):
%Script that builds a linear model from a nonlinear PST model
%
%Author: RJ Hallett
%Date: 2017
% clear; clc; close all;
PSTpath = strcat(pwd,'\pstV2p2');
addpath(PSTpath)
copyfile('d_microWECC_V2_thesis.m','DataFile.m')
svm_mgen_Batch
save LinModel_microV4_test
```

29. Appendix T: MicroWECC Eigenanalysis Code

```
%% Conduct modal analysis of microWECC_V2

%Authors: Dan Trudnowski, 2015; RJ Hallett, 2018
clear all; clc
close all;
%% Settings
Eig.Num = 1; %Index of Eig.Desired to analyze.  See list below.
Eig.Desired = NaN(33,1); %Set modes (If(isnan), use default in Eig.Default).
See list below.
% Eig.Desired(2) = [(-1/tan(acos(0.02)) + 1j)*0.3719*2*pi]; Set to a desired
value

Eig.File = 'LinModel_microV2_thesis'; %Case to analyze

%% Default eigenvalues in the electromechanical range and the inertia
constants
load(Eig.File,'l');
F = [0.1 2];
x = l(imag(l)>F(1)*2*pi & imag(l)<F(2)*2*pi & -100*real(l)./abs(l)<20);
[~,n] = sort(imag(x));
Eig.Default = x(n);
clear x n F
% [[1:length(Eig.Default)]' imag(Eig.Default)./(2*pi) -
100*real(Eig.Default)./abs(Eig.Default) Eig.Default]
%1    0.1929           17.0785           -0.2100 + 1.2118i %NSA
%2    0.3499            4.0094           -0.0882 + 2.1983i %NSB
%3    0.5131           10.4526           -0.3388 + 3.2237i %Gen 4 and 9 vs
sys (EWA)
%4    0.7240           11.4246           -0.5232 + 4.5493i %Gen 1 vs sys
(MT)
%5    0.7430            9.7162           -0.4557 + 4.6684i %Gen 1 vs sys
%6    0.8475            6.6900           -0.3570 + 5.3249i %Gen 3 and 8 vs
sys
%7    0.9321            6.6137           -0.3385 + 5.8562i %Gen 3
%8    1.1615           11.9010           -0.8748 + 7.2980i %Gen 5 vs 6
eval('d_microWECC_V10_4_test_thesis');
H = mac_con(:,16).*mac_con(:,3)./100; %Interia constants on system base
save delme H Eig
clear all
load delme
delete delme.mat
%% Load A matrix
load(Eig.File,'u','v','l','p','ang_idx');

[~,ModeNum] = min(abs(l-Eig.Default(Eig.Num)));
Mode = [imag(l(ModeNum))/(2*pi) -100*real(l(ModeNum))/abs(l(ModeNum))];
ModeEig = l(ModeNum);
%% Analysis
Index = ang_idx + 1; %Index for gen 1-34 speed states
Part=abs(p(Index,ModeNum)); Part=Part./max(abs(Part)); %participation for
gens 1-34 speed states
Part_H = Part./H; Part_H = Part_H./max(abs(Part_H)); %participation
normalized by inertia
```

```
Modeshape=(180/pi)*angle(u(Index, ModeNum));  %mode shape angle
ModeAmp = abs(u(Index, ModeNum))./max(abs(u(Index, ModeNum))); %Mode
amplitudes
x=[[1:9]' Part ModeAmp Modeshape]; [xx,i]=sort(x(:,3).^(-1)); x=x(i,:);
x(:,4)=x(:,4)-x(1,4);
x=x(x(:,3)>=0.1 ,:); %Keep Gens with amplitude >= 0.1
%% Display
n0 = find(abs(x(:,4))<=70); %Index for machines in phase
n180 = find(abs(x(:,4))>70 & abs(x(:,4))<=230); %Index for machines out of
phase
n360 = find(abs(x(:,4))>230);

disp('    Freq(Hz)   Damping(%)')
disp(Mode(1,:))
disp('Gen  Part    Amplitude Shape(deg)')
for k=1:length(n0);

disp(sprintf('%3.0f\t%4.3f\t%4.3f\t%4.0f',x(n0(k),1),x(n0(k),2),x(n0(k),3),x(
n0(k),4)));
end
for k=1:length(n180);

disp(sprintf('%3.0f\t%4.3f\t%4.3f\t%4.0f',x(n180(k),1),x(n180(k),2),x(n180(k)
,3),x(n180(k),4)));
end
for k=1:length(n360);

disp(sprintf('%3.0f\t%4.3f\t%4.3f\t%4.0f',x(n360(k),1),x(n360(k),2),x(n360(k)
,3),x(n360(k),4)));
end
```

30. Appendix U: MicroWECC and MiniWECC Eigenanalysis Results

Freq(Hz)	Damping(%)
0.1929	17.0785

Gen	Part	Amplitude	Shape(deg)
2	1.000	1.000	0
4	0.081	0.382	-25
3	0.020	0.217	-45
1	0.006	0.192	-58
5	0.010	0.182	-59
6	0.016	0.179	-62
9	0.310	0.374	-121
8	0.254	0.340	-125
7	0.026	0.179	-139

Figure 88. MicroWECC NSA Mode

Freq(Hz)	Damping(%)
0.2183	5.6288

Gen	Part	Amplitude	Shape(deg)
34	1.000	1.000	0
3	0.020	0.294	9
1	0.013	0.220	0
2	0.010	0.206	5
12	0.000	0.130	13
30	0.035	0.218	-158
26	0.017	0.214	-156
27	0.041	0.213	-156
28	0.035	0.211	-157
23	0.013	0.210	-160
25	0.017	0.204	-156
24	0.008	0.203	-155
22	0.008	0.198	-159
21	0.046	0.179	-160
19	0.008	0.169	-158
20	0.006	0.167	-156
29	0.021	0.164	-158
31	0.003	0.127	-158
33	0.001	0.106	-158

Figure 89. MiniWECC NSA Mode

Freq(Hz)	Damping(%)
0.3499	4.0094

Gen	Part	Amplitude	Shape(deg)
9	1.000	1.000	0
2	0.574	0.704	-51
8	0.313	0.648	-2
4	0.589	0.876	109
3	0.320	0.736	130
5	0.194	0.668	136
1	0.112	0.663	124
6	0.322	0.662	136
7	0.082	0.338	159

Figure 90. MicroWECC NSB Mode

Freq(Hz)	Damping(%)
0.3716	3.7834

Gen	Part	Amplitude	Shape(deg)
1	0.592	1.000	0
2	0.368	0.882	5
14	0.277	0.845	-5
13	0.164	0.830	0
7	0.638	0.816	6
5	0.369	0.815	3
4	0.156	0.792	5
12	0.026	0.774	4
8	0.229	0.721	9
9	0.095	0.705	7
10	0.294	0.698	8
6	0.075	0.687	6
11	0.205	0.679	7
3	0.194	0.645	2
15	0.031	0.601	8
16	0.037	0.326	11
17	0.015	0.216	-2
18	0.031	0.214	-2
32	0.008	0.167	-9
33	0.003	0.124	-13
34	1.000	0.695	-171
23	0.084	0.521	-161
26	0.157	0.511	-155
27	0.337	0.508	-155
28	0.302	0.495	-155
30	0.307	0.490	-160
24	0.068	0.468	-153
25	0.121	0.458	-154
22	0.038	0.416	-156
21	0.157	0.285	-155
19	0.019	0.215	-148
20	0.013	0.197	-145
29	0.045	0.173	-147

Figure 91. MiniWECC NSB Mode

Freq(Hz)	Damping(%)
0.5131	10.4526

Gen	Part	Amplitude	Shape(deg)
4	1.000	1.000	0
9	0.044	0.230	-30
7	0.090	0.350	162
6	0.114	0.349	154
1	0.043	0.346	133
5	0.065	0.339	154
3	0.054	0.267	133
2	0.024	0.126	174

Figure 92. MicroWECC BC Mode

Freq(Hz)	Damping(%)
0.6239	5.6980

Gen	Part	Amplitude	Shape(deg)
1	1.000	1.000	0
2	0.282	0.604	17
3	0.130	0.402	27
23	0.013	0.192	-1
26	0.031	0.189	29
27	0.065	0.188	30
28	0.052	0.171	29
30	0.030	0.141	16
24	0.008	0.133	31
25	0.013	0.126	30
14	0.138	0.468	147
32	0.050	0.398	206
13	0.047	0.355	159
31	0.062	0.340	203
16	0.035	0.283	206
17	0.026	0.270	206
33	0.014	0.264	201
18	0.051	0.259	207
8	0.043	0.257	201
10	0.056	0.252	202
15	0.007	0.247	204
6	0.013	0.238	203
9	0.015	0.233	201
11	0.034	0.231	202
5	0.033	0.198	201
29	0.049	0.168	203
7	0.036	0.161	206
12	0.001	0.126	173
20	0.007	0.119	213

Figure 93. MiniWECC BC Mode

Freq(Hz)	Damping(%)
0.7240	11.4246

Gen	Part	Amplitude	Shape(deg)
1	1.000	1.000	0
9	0.119	0.184	60
7	0.120	0.226	217
6	0.041	0.128	157
5	0.018	0.108	145
8	0.210	0.294	250

Figure 94. MicroWECC MT Mode

Freq(Hz)	Damping(%)
0.6867	9.6153

Gen	Part	Amplitude	Shape(deg)
14	1.000	1.000	0
13	0.245	0.636	12
12	0.002	0.159	48
23	0.012	0.149	-26
27	0.059	0.144	43
26	0.026	0.139	37
28	0.046	0.127	41
30	0.027	0.101	-48
32	0.038	0.300	176
31	0.045	0.241	178
17	0.020	0.192	203
33	0.011	0.189	173
18	0.038	0.182	203
16	0.021	0.182	191
5	0.040	0.176	156
29	0.062	0.152	176
15	0.004	0.145	165
6	0.007	0.142	157
10	0.022	0.129	146
11	0.014	0.118	148
8	0.014	0.117	129
9	0.005	0.103	131
4	0.006	0.100	156
1	0.090	0.234	245
20	0.018	0.153	244
19	0.016	0.130	246
21	0.062	0.117	255

Figure 95. MiniWECC MT Mode

31. Appendix V: PST and PSLF Hydro Governor Comparison

```
close all;
clc;
clear;

R  = 0.05;  %Droop

Ts =  0.5;
Tc =   15;
T3 =  1.0;
T4 = -1.0;
T5 =  0.5;

T_1 =  0.2;
T_2 =   -1;
T_3 = 15.0;

sim('gov.slx')

figure
hold on
plot(tout,yout(:,1),'k--','linewidth',4)
plot(tout,yout(:,2),'r','linewidth', 2)
title('Hydro Turbine-Gov Step Response')
xlabel('Time (s)')
ylabel('Output')
legend('PST Model','PSLF Model','location','best')
axis([0 10 -0.2 0.5])
grid on
ax = gca;
ax.GridAlpha = 1;
set(gca,'gridlinestyle','-');
```

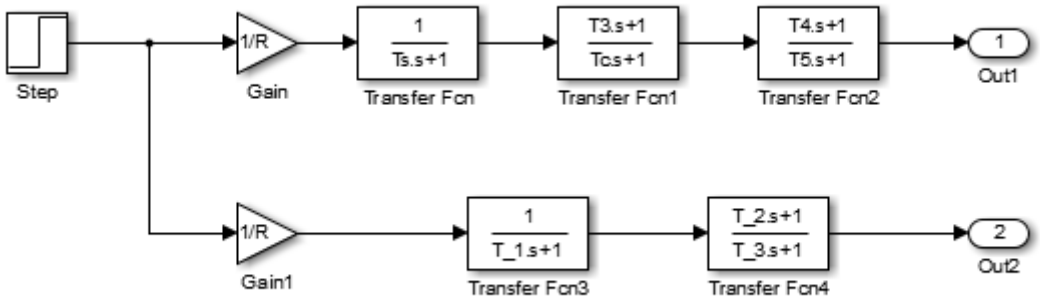

Figure 96. Simulink Model of PST and PSLF Hydro Generators

SIGNATURE PAGE

This is to certify that the thesis prepared by Robin Hallett entitled "Improving a Transient Stability Control Scheme with Wide-Area Synchrophasors and The MicroWECC Model, A Reduced-Order Model of the Western Interconnect" has been examined and approved for acceptance by the Department of Electrical Engineering, Montana Tech of The University of Montana, on this 7th day of December, 2018.

Dan Trudnowski, PhD, Dean of the School of Mines and Engineering
Department of Electrical Engineering
Chair, Examination Committee

Josh Wold, PhD, Assistant Professor
Department of Electrical Engineering
Member, Examination Committee

Curtis Link, PhD, Professor Emeritus
Department of Geophysical Engineering
Member, Examination Committee

Eric Bahr, Regional Transmission Planning Engineer
Northwestern Energy
Member, Examination Committee